ATARI
400/800
OPERATING
SYSTEM

why doesn't upper RAM clear
on Dev. Systems?

OWING VECTOR

```
                        ;
                        ;       COLLEEN OPERATING SYSTEM EQUATE FILE
                        ;
                        ;
                        ;
                        ;       MODULE ORIGIN TABLE
                        ;
     E000               CHRORG  = $E000 CHARACTER SET
     E400               VECTBL  = $E400 VECTOR TABLE
     E480               VCTABL  = $E480 RAM VECTOR INITIAL VALUE TABLE
     E4A6               CIOORG  = $E4A6 CENTRAL I/O HANDLER
     E6D5               INTORG  = $E6D5 INTERRUPT HANDLER
     E944               SIOORG  = $E944 SERIAL I/O DRIVER
     EDEA               DSKORG  = $EDEA DISK HANDLER
     EE78               PRNORG  = $EE78 PRINTER HANDLER
     EF41               CASORG  = $EF41 CASSETTE HANDLER
     F0E3               MONORG  = $F0E3 MONITOR/POWER UP MODULE
     F3E4               KBDORG  = $F3E4 KEYBOARD/DISPLAY HANDLER
                        ;
                        ;
                        ;
                        ;
                        ;       VECTOR TABLE
                        ;
                        ;HANDLER ENTRY POINTS ARE CALLED OUT IN THE FOLLOWING VECTOR
                        ;TABLE. THESE ARE THE ADDRESSES MINUS ONE.
                        ;
                        ;
                        ;EXAMPLE FOR EDITOR
                        ;
                        ;    E400       OPEN
                        ;       2       CLOSE
                        ;       4       GET
                        ;       6       PUT
                        ;       8       STATUS
                        ;       A       SPECIAL
                        ;       C       JUMP TO POWER ON INITIALIZATION ROUTINE
                        ;       E       NOT USED
                        ;
                        ;
     E400               EDITRV  = $E400 EDITOR
     E410               SCRENV  = $E410 TELEVISION SCREEN
     E420               KEYBDV  = $E420 KEYBOARD
     E430               PRINTV  = $E430 PRINTER
     E440               CASETV  = $E440 CASSETTE
                        ;
                        ;       JUMP VECTOR TABLE
                        ;
                        ;THE FOLLOWING IS A TABLE OF JUMP INSTRUCTIONS
                        ;TO VARIOUS ENTRY POINTS IN THE OPERATING SYSTEM.
                        ;
     E450               DISKIV  = $E450 DISK INITIALIZATION
     E453               DSKINV  = $E453 DISK INTERFACE
     E456               CIOV    = $E456 CENTRAL INPUT OUTPUT ROUTINE
     E459               SIOV    = $E459 SERIAL INPUT OUTPUT ROUTINE
     E45C               SETVBV  = $E45C SET SYSTEM TIMERS ROUTINE
     E45F               SYSVBV  = $E45F SYSTEM VERTICAL BLANK CALCULATIONS
     E462               XITVBV  = $E462 EXIT VERTICAL BLANK CALCULATIONS
```

[illegible]

```
CENTRAL INPUT/OUTPUT (CIO) 2-7-79

     E465             SIOINV = $E465 SERIAL INPUT OUTPUT INITIALIZATION
     E468             SENDEV = $E468 SEND ENABLE ROUTINE
     E46B             INTINV = $E46B INTERRUPT HANDLER INITIALIZATION
     E46E             CIOINV = $E46E CENTRAL INPUT OUTPUT INITIALIZATION
     E471             BLKBDV = $E471 BLACKBOARD MODE
     E474             WARMSV = $E474 WARM START ENTRY POINT
     E477             COLDSV = $E477 COLD START ENTRY POINT
     E47A             RBLOKV = $E47A CASSETTE READ BLOCK ENTRY POINT VECTOR
     E47D             CSOPIV = $E47D CASSETTE OPEN FOR INPUT VECTOR
     E480             VCTABL = $E480

                      ;
                      ;
                      ; OPERATING SYSTEM EQUATES
                      ;
                      ; COMMAND CODES FOR IOCB
     0003             OPEN   = 3               OPEN FOR INPUT/OUTPUT
     0005             GETREC = 5               GET RECORD (TEXT)
     0007             GETCHR = 7               GET CHARACTER(S)
     0009             PUTREC = 9               PUT RECORD (TEXT)
     000B             PUTCHR = $B              PUT CHARACTER(S)
     000C             CLOSE  = $C              CLOSE DEVICE
     000D             STATIS = $D              STATUS REQUEST
     000E             SPECIL = $E              BEGINNING OF SPECIAL ENTRY COMMANDS
                      ;
                      ; SPECIAL ENTRY COMMANDS
     0011             DRAWLN = $11             DRAW LINE
     0012             FILLIN = $12             DRAW LINE WITH RIGHT FILL
     0020             RENAME = $20             RENAME DISK FILE
     0021             DELETE = $21             DELETE DISK FILE
     0022             FORMAT = $22             FORMAT
     0023             LOCKFL = $23             LOCK FILE TO READ ONLY
     0024             UNLOCK = $24             UNLOCK LOCKED FILE
     0025             POINT  = $25             POINT SECTOR
     0026             NOTE   = $26             NOTE SECTOR
     00FF             IOCFRE = $FF             IOCB "FREE"
                      ;
                      ; AUX1 EQUATES
                      ; () INDICATES WHICH DEVICES USE BIT
     0001             APPEND = $1              OPEN FOR WRITE APPEND (D), OR SCREEN READ (E)
     0002             DIRECT = $2              OPEN FOR DIRECTORY ACCESS (D)
     0004             OPNIN  = $4              OPEN FOR INPUT (ALL DEVICES)
     0008             OPNOT  = $8              OPEN FOR OUTPUT (ALL DEVICES)
     000C             OPNINO = OPNIN+OPNOT     OPEN FOR INPUT AND OUTPUT (ALL DEVICES)
     0010             MXDMOD = $10             OPEN FOR MIXED MODE (E,S)
     0020             INSCLR = $20             OPEN WITHOUT CLEARING SCREEN (E,S)
                      ;
                      ; DEVICE NAMES
     0045             SCREDT = 'E              SCREEN EDITOR (R/W)
     004B             KBD    = 'K              KEYBOARD (R ONLY)
     0053             DISPLY = 'S              SCREEN DISPLAY (R/W)
     0050             PRINTR = 'P              PRINTER (W ONLY)
     0043             CASSET = 'C              CASSETTE
     004D             MODEM  = 'M              MODEM
     0044             DISK   = 'D              DISK (R/W)
                      ;
                      ; SYSTEM EOL (CARRIAGE RETURN)
     009B             CR     = $9B
```

```
                      ;        OPERATING SYSTEM STATUS CODES
                      ;
    0001              SUCCES = $01    SUCCESSFUL OPERATION
                      ;
    0080              BRKABT = $80    BREAK KEY ABORT
    0081              PRVOPN = $81    IOCB ALREADY OPEN
    0082              NONDEV = $82    NON-EXISTANT DEVICE
    0083              WRONLY = $83    IOCB OPENED FOR WRITE ONLY
    0084              NVALID = $84    INVALID COMMAND
    0085              NOTOPN = $85    DEVICE OR FILE NOT OPEN
    0086              BADIOC = $86    INVALID IOCB NUMBER
    0087              RDONLY = $87    IOCB OPENED FOR READ ONLY
    0088              EOFERR = $88    END OF FILE
    0089              TRNRCD = $89    TRUNCATED RECORD
    008A              TIMOUT = $8A    PERIPHERAL DEVICE TIME OUT
    008B              DNACK  = $8B    DEVICE DOES NOT ACKNOWLEDGE COMMAND
    008C              FRMERR = $8C    SERIAL BUS FRAMING ERROR
    008D              CRSROR = $8D    CURSOR OVERRANGE
    008E              OVRRUN = $8E    SERIAL BUS DATA OVERRUN
    008F              CHKERR = $8F    SERIAL BUS CHECKSUM ERROR
                      ;
    0090              DERROR = $90    PERIPHERAL DEVICE ERROR (OPERATION NOT COMPLETED)
    0091              BADMOD = $91    BAD SCREEN MODE NUMBER
    0092              FNCNOT = $92    FUNCTION NOT IMPLEMENTED IN HANDLER
    0093              SCRMEM = $93    INSUFICIENT MEMORY FOR SCREEN MODE
                      ;
                      ;
                      ;
                      ;
                      ;
                      ;
                      ;        PAGE ZERO RAM ASSIGNMENTS
                      ;
                               *=$0000
    0000              LINZBS *=*+2    LINBUG RAM (WILL BE REPLACED BY MONITOR RAM)
                      ;
                      ; THESE LOCATIONS ARE NOT CLEARED
    0002              CASINI *=*+2    CASSETTE INIT LOCATION
    0004              RAMLO  *=*+2    RAM POINTER FOR MEMORY TEST
    0006              TRAMSZ *=*+1    TEMPORARY REGISTER FOR RAM SIZE
    0007              TSTDAT *=*+1    RAM TEST DATA REGISTER
                      ;
                      ; CLEARED ON COLDSTART ONLY
    0008              WARMST *=*+1    WARM START FLAG
    0009              BOOT?  *=*+1    SUCCESSFUL BOOT FLAG
    000A              DOSVEC *=*+2    DISK SOFTWARE START VECTOR
    000C              DOSINI *=*+2    DISK SOFTWARE INIT ADDRESS
    000E              APPMHI *=*+2    APPLICATIONS MEMORY HI LIMIT
                      ;
                      ; CLEARED ON COLD OR WARM START
    0010              INTZBS =*       INTERRUPT HANDLER
    0010              POKMSK *=*+1    SYSTEM MASK FOR POKEY IRQ ENABLE
    0011              BRKKEY *=*+1    BREAK KEY FLAG
    0012              RTCLOK *=*+3    REAL TIME CLOCK (IN 16 MSEC UNITS)
                      ;
    0015              BUFADR *=*+2    INDIRECT BUFFER ADDRESS REGISTER
                      ;
    0017              ICCOMT *=*+1    COMMAND FOR VECTOR
                      ;
    0018              DSKFMS *=*+2    DISK FILE MANAGER POINTER
```

```
                              DISK FILE MANAGER POINTER

CENTRAL INPUT/OUTPUT (CIO) 2-7-79

001A          DSKUTL  *=*+2    DISK UTILITIES POINTER
                ;
001C          PTIMOT  *=*+1    PRINTER TIME OUT REGISTER
001D          PBPNT   *=*+1    PRINT BUFFER POINTER
001E          PBUFSZ  *=*+1    PRINT BUFFER SIZE
001F          PTEMP   *=*+1    TEMPORARY REGISTER
                ;
0020          ZIOCB   =*       ZERO PAGE I/O CONTROL BLOCK
0010          IOCBSZ  = 16     NUMBER OF BYTES PER IOCB
0080          MAXIOC  = 8*IOCBSZ      LENGTH OF THE IOCB AREA
0020          IOCBAS  =*
0020          ICHIDZ  *=*+1    HANDLER INDEX NUMBER (FF = IOCB FREE)
0021          ICDNOZ  *=*+1    DEVICE NUMBER (DRIVE NUMBER)
0022          ICCOMZ  *=*+1    COMMAND CODE
0023          ICSTAZ  *=*+1    STATUS OF LAST IOCB ACTION
0024          ICBALZ  *=*+1    BUFFER ADDRESS LOW BYTE
0025          ICBAHZ  *=*+1
0026          ICPTLZ  *=*+1    PUT BYTE ROUTINE ADDRESS - 1
0027          ICPTHZ  *=*+1
0028          ICBLLZ  *=*+1    BUFFER LENGTH LOW BYTE
0029          ICBLHZ  *=*+1
002A          ICAX1Z  *=*+1    AUXILIARY INFORMATION FIRST BYTE
002B          ICAX2Z  *=*+1
002C          ICSPRZ  *=*+4    TWO SPARE BYTES (CIO LOCAL USE)
002E          ICIDNO  = ICSPRZ+2  IOCB NUMBER X 16
002F          CIOCHR  = ICSPRZ+3  CHARACTER BYTE FOR CURRENT OPERATION
                ;
0030          STATUS  *=*+1    INTERNAL STATUS STORAGE
0031          CHKSUM  *=*+1    CHECKSUM (SINGLE BYTE SUM WITH CARRY)
0032          BUFRLO  *=*+1    POINTER TO DATA BUFFER (LO BYTE)
0033          BUFRHI  *=*+1    POINTER TO DATA BUFFER (HI BYTE)
0034          BFENLO  *=*+1    NEXT BYTE PAST END OF THE DATA BUFFER (LO BYTE)
0035          BFENHI  *=*+1    NEXT BYTE PAST END OF THE DATA BUFFER (HI BYTE)
0036          CRETRY  *=*+1    NUMBER OF COMMAND FRAME RETRIES
0037          DRETRY  *=*+1    NUMBER OF DEVICE RETRIES                        SIO
0038          BUFRFL  *=*+1    DATA BUFFER FULL FLAG
0039          RECVDN  *=*+1    RECEIVE DONE FLAG
003A          XMTDON  *=*+1    TRANSMISSION DONE FLAG
003B          CHKSNT  *=*+1    CHECKSUM SENT FLAG
003C          NOCKSM  *=*+1    NO CHECKSUM FOLLOWS DATA FLAG
                ;
003D          BPTR    *=*+1    current buffer index
003E          FTYPE   *=*+1    [illegible]                                     cassette handler
003F          FEOF    *=*+1    eof flag
0040          FREQ    *=*+1                                                    Sound I/O ?
0041          SOUNDR  *=*+1    NOISY I/O FLAG.  (ZERO IS QUIET)
0042          CRITIC  *=*+1    DEFINES CRITICAL SECTION (CRITICAL IF NON-ZERO) ← SIO, ...
                ;
0043          FMSZPG  *=*+7    DISK FILE MANAGER SYSTEM ZERO PAGE
                ;
                ;
004A          CKEY    *=*+1    FLAG SET WHEN GAME START PRESSED
004B          CASSBT  *=*+1    CASSETTE BOOT FLAG
004C          DSTAT   *=*+1    DISPLAY STATUS
                ;
004D          ATRACT  *=*+1    ATRACT FLAG
004E          DRKMSK  *=*+1    DARK ATRACT MASK
004F          COLRSH  *=*+1    ATRACT COLOR SHIFTER (EOR'ED WITH PLAYFIELD COLORS)
                ;
```

```
0002                LEDGE   =  2        LMARGN'S VALUE AT COLD START
0027                REDGE   =  39       RMARGN'S VALUE AT COLD START
0050                TMPCHR  *=*+1
0051                HOLD1   *=*+1
0052                LMARGN  *=*+1       LEFT MARGIN (SET TO 1 AT POWER ON)
0053                RMARGN  *=*+1       RIGHT MARGIN (SET TO 38 AT POWER ON)
0054                ROWCRS  *=*+1       CURSOR COUNTERS
0055                COLCRS  *=*+2
0057                DINDEX  *=*+1
0058                SAVMSC  *=*+2       [illegible]
005A                OLDROW  *=*+1
005B                OLDCOL  *=*+2
005D                OLDCHR  *=*+1       DATA UNDER CURSOR
005E                OLDADR  *=*+2
0060                NEWROW  *=*+1       POINT DRAW GOES TO
0061                NEWCOL  *=*+2
0063                LOGCOL  *=*+1       POINTS AT COLUMN IN LOGICAL LINE
0064                ADRESS  *=*+2
0066                MLTTMP  *=*+2
0066                OPNTMP  =  MLTTMP         FIRST BYTE IS USED IN OPEN AS TEMP
0068                SAVADR  *=*+2
006A                RAMTOP  *=*+1       RAM SIZE DEFINED BY POWER ON LOGIC
006B                BUFCNT  *=*+1       BUFFER COUNT
006C                BUFSTR  *=*+2       EDITOR GETCH POINTER
006E                BITMSK  *=*+1       BIT MASK
006F                SHFAMT  *=*+1
0070                ROWAC   *=*+2
0072                COLAC   *=*+2
0074                ENDPT   *=*+2
0076                DELTAR  *=*+1
0077                DELTAC  *=*+2
0079                ROWINC  *=*+1
007A                COLINC  *=*+1
007B                SWPFLG  *=*+1       NON-0 IF TXT AND REGULAR RAM IS SWAPPED
007C                HOLDCH  *=*+1       CH IS MOVED HERE IN KGETCH BEFORE CNTL & SHIFT PROC
007D                INSDAT  *=*+1
007E                COUNTR  *=*+2
                    ;
                    ;
                    ;
                    ;       80 - FF ARE RESERVED FOR USER APPLICATIONS
                    ;
                    ;
                    ;
                    ;       NOTE : SEE FLOATING POINT SUBROUTINE AREA FOR ZERO PAGE CELLS
                    ;
                    ;
                    ;
                    ;
                    ;       PAGE 1  -  STACK
                    ;
                    ;
                    ;
                    ;
                    ;       PAGE TWO RAM ASSIGNMENTS
                    ;
                            *=$0200
0200                INTABS  =*          INTERRUPT RAM
0200                VDSLST  *=*+2       DISPLAY LIST NMI VECTOR
                    ;
```

```
0202            VPRCED  *=*+2   PROCEED LINE IRQ VECTOR
0204            VINTER  *=*+2   INTERRUPT LINE IRQ VECTOR
0206            VBREAK  *=*+2   SOFTWARE BREAK (00) INSTRUCTION IRQ VECTOR
0208            VKEYBD  *=*+2   POKEY KEYBOARD IRQ VECTOR
020A            VSERIN  *=*+2   POKEY SERIAL INPUT READY IRQ
020C            VSEROR  *=*+2   POKEY SERIAL OUTPUT READY IRQ
020E            VSEROC  *=*+2   POKEY SERIAL OUTPUT COMPLETE IRQ
0210            VTIMR1  *=*+2   POKEY TIMER 1 IRQ
0212            VTIMR2  *=*+2   POKEY TIMER 2 IRQ
0214            VTIMR4  *=*+2   POKEY TIMER 4 IRQ
0216            VIMIRQ  *=*+2   IMMEDIATE IRQ VECTOR
0218            CDTMV1  *=*+2   COUNT DOWN TIMER 1
021A            CDTMV2  *=*+2   COUNT DOWN TIMER2
021C            CDTMV3  *=*+2   COUNT DOWN TIMER 3
021E            CDTMV4  *=*+2   COUNT DOWN TIMER 4
0220            CDTMV5  *=*+2   COUNT DOWN TIMER 5
0222            VVBLKI  *=*+2   IMMEDIATE VERTICAL BLANK NMI VECTOR
0224            VVBLKD  *=*+2   DEFERRED VERTICAL BLANK NMI VECTOR
0226            CDTMA1  *=*+2   COUNT DOWN TIMER 1 JSR ADDRESS ✓
0228            CDTMA2  *=*+2   COUNT DOWN TIMER 2 JSR ADDRESS ✓
022A            CDTMF3  *=*+1   COUNT DOWN TIMER 3 FLAG ✓
022B           -SRTIMR  *=*+1   SOFTWARE REPEAT TIMER
022C            CDTMF4  *=*+1   COUNT DOWN TIMER 4 FLAG ✓
022D            INTEMP  *=*+1   IAN'S TEMP (RENAMED FROM T1 BY POPULAR DEMAND)
022E            CDTMF5  *=*+1   COUNT DOWN TIMER FLAG 5 ✓
022F            SDMCTL  *=*+1   SAVE DMACTL REGISTER
0230            SDLSTL  *=*+1   SAVE DISPLAY LIST LOW BYTE
0231            SDLSTH  *=*+1   SAVE DISPLAY LIST HI BYTE
0232            SSKCTL  *=*+1   SKCTL REGISTER RAM
                        *=*+1
                ;
0234           -LPENH   *=*+1   LIGHT PEN HORIZONTAL VALUE
0235           -LPENV   *=*+1   LIGHT PEN VERTICAL VALUE
                        *=*+2   SPARE BYTES
                ;
                        *=*+2   SPARE
                ;
023A           -CDEVIC  *=*+1   COMMAND FRAME BUFFER - DEVICE
023B           -CCOMND  *=*+1                          COMMAND
023C           -CAUX1   *=*+1                           COMMAND AUX BYTE 1
023D           -CAUX2   *=*+1                           COMMAND AUX BYTE 2
                ;               NOTE: MAY NOT BE THE LAST WORD ON A PAGE
023E            TEMP    *=*+1   TEMPORARY RAM CELL
                ;               NOTE: MAY NOT BE THE LAST WORD ON A PAGE
023F            ERRFLG  *=*+1   ERROR FLAG - ANY DEVICE ERROR EXCEPT TIME OUT
                ;
0240            DFLAGS  *=*+1   DISK FLAGS FROM SECTOR ONE
0241            DBSECT  *=*+1   NUMBER OF DISK BOOT SECTORS
0242            BOOTAD  *=*+2   ADDRESS WHERE DISK BOOT LOADER WILL BE PUT
0244            COLDST  *=*+1   COLDSTART FLAG (1=IN MIDDLE OF COLDSTART)
                ;
                        *=*+1   SPARE
                ;
0246            DSKTIM  *=*+1   DISK TIME OUT REGISTER
                ;
0247            LINBUF  *=*+40  CHAR LINE BUFFER
                ;
026F            GPRIOR  *=*+1   GLOBAL PRIORITY CELL
                ;
0270           -PADDL0  *=*+1   POTENTIOMETER 0 RAM CELL
```

```
CENTRAL INPUT/OUTPUT (CIO) [illegible]

 0271      PADDL1  *=*+1
 0272      PADDL2  *=*+1
 0273      PADDL3  *=*+1
 0274      PADDL4  *=*+1
 0275      PADDL5  *=*+1
 0276      PADDL6  *=*+1
 0277      PADDL7  *=*+1
 0278      STICK0  *=*+1   JOYSTICK 0 RAM CELL
 0279      STICK1  *=*+1
 027A      STICK2  *=*+1
 027B      STICK3  *=*+1
 027C      PTRIG0  *=*+1   PADDLE TRIGGER 0
 027D      PTRIG1  *=*+1
 027E      PTRIG2  *=*+1
 027F      PTRIG3  *=*+1
 0280      PTRIG4  *=*+1
 0281      PTRIG5  *=*+1
 0282      PTRIG6  *=*+1
 0283      PTRIG7  *=*+1
 0284      STRIG0  *=*+1   JOYSTICK TRIGGER 0
 0285      STRIG1  *=*+1
 0286      STRIG2  *=*+1
 0287      STRIG3  *=*+1

 0288      CSTAT   *=*+1                      [illegible]
 0289     -WMODE   *=*+1   [illegible]
 028A     -BLIM    *=*+1   [illegible]

                   *=*+5   SPARE



 0290     -TXTROW  *=*+1   TEXT ROWCRS
 0291     -TXTCOL  *=*+2   TEXT COLCRS
 0293      TINDEX  *=*+1   TEXT INDEX
 0294      TXTMSC  *=*+2   FOOLS CONVRT INTO NEW MSC
 0296     -TXTOLD  *=*+6   OLDROW & OLDCOL FOR TEXT (AND THEN SOME)
 029C      TMPX1   *=*+1
 029D      HOLD3   *=*+1
 029E      SUBTMP  *=*+1
 029F      HOLD2   *=*+1
 02A0      DMASK   *=*+1
 02A1      TMPLBT  *=*+1
 02A2     -ESCFLG  *=*+1   ESCAPE FLAG
 02A3     -TABMAP  *=*+15
 02B2     -LOGMAP  *=*+4   LOGICAL LINE START BIT MAP
 02B6     -INVFLG  *=*+1   INVERSE VIDEO FLAG (TOGGLED BY ATARI KEY)
 02B7     -FILFLG  *=*+1   RIGHT FILL FLAG FOR DRAW
 02B8     -TMPROW  *=*+1
 02B9      TMPCOL  *=*+2
 02BB     -SCRFLG  *=*+1   SET IF SCROLL OCCURS
 02BC     -HOLD4   *=*+1   TEMP CELL USED IN DRAW ONLY
 02BD     -HOLD5   *=*+1   DITTO
 02BE     -SHFAMT  *=*+1
 02BF     -BOTSCR  *=*+1   BOTTOM OF SCREEN:  24 NORM,  4 SPLIT


 02C0      PCOLR0  *=*+1   P0 COLOR
 02C1      PCOLR1  *=*+1   P1 COLOR
```

```
  02C1              PCOLR1  *=*+1   P1 COLOR
```

```
  02C2              PCOLR2  *=*+1   P2 COLOR
  02C3              PCOLR3  *=*+1   P3 COLOR
  02C4              COLOR0  *=*+1   COLOR 0
  02C5              COLOR1  *=*+1
  02C6              COLOR2  *=*+1
  02C7              COLOR3  *=*+1
  02C8              COLOR4  *=*+1
                    ;
                    ;
                            *=*+23  SPARE
                    ;
                    ;
                    ;
  02E0              GLBABS  *=*     GLOBAL VARIABLES
                    ;
                            *=*+4   SPARE
                    ;
  02E4              RAMSIZ  *=*+1   RAM SIZE (HI BYTE ONLY)
  02E5              MEMTOP  *=*+2   TOP OF AVAILABLE USER MEMORY
  02E7              MEMLO   *=*+2   BOTTOM OF AVAILABLE USER MEMORY
                            *=*+1   SPARE
  02EA              DVSTAT  *=*+4   STATUS BUFFER
  02EE              CBAUDL  *=*+1   CASSETTE BAUD RATE LOW BYTE
  02EF              CBAUDH  *=*+1
                    ;
  02F0              CRSINH  *=*+1   CURSOR INHIBIT (00 = CURSOR ON)
  02F1              KEYDEL  *=*+1   KEY DELAY
  02F2              CH1     *=*+1
                    ;
  02F3              CHACT   *=*+1   CHACTL REGISTER RAM
  02F4              CHBAS   *=*+1   CHBAS REGISTER RAM
                    ;
                            *=*+5   SPARE BYTES
                    ;
  02FA          ?   CHAR    *=*+1   [illegible]
  02FB              ATACHR  *=*+1   ATASCII CHARACTER
  02FC              CH      *=*+1   GLOBAL VARIABLE FOR KEYBOARD
  02FD              FILDAT  *=*+1   RIGHT FILL DATA (DRAW)
  02FE              DSPFLG  *=*+1   DISPLAY FLAG : DISPLAY CNTLS IF NON-ZERO
  02FF              SSFLAG  *=*+1   START/STOP FLAG FOR PAGING (CNTL 1). CLEARED BY BREAK
                    ;
                    ;
                    ;
                    ;
                    ;
                    ;
                    ;       PAGE THREE RAM ASSIGNMENTS
                    ;
  0300              DCB     *=*     DEVICE CONTROL BLOCK
  0300              DDEVIC  *=*+1   PERIPHERAL UNIT 1 BUS I.D. NUMBER
  0301              DUNIT   *=*+1   UNIT NUMBER
  0302              DCOMND  *=*+1   BUS COMMAND
  0303              DSTATS  *=*+1   COMMAND TYPE/STATUS RETURN
  0304              DBUFLO  *=*+1   DATA BUFFER POINTER LOW BYTE
  0305              DBUFHI  *=*+1
  0306              DTIMLO  *=*+1   DEVICE TIME OUT IN 1 SECOND UNITS
  0307              DUNUSE  *=*+1   UNUSED BYTE
  0308              DBYTLO  *=*+1   NUMBER OF BYTES TO BE TRANSFERRED LOW BYTE
  0309              DBYTHI  *=*+1
```

```
030A          DAUX1   *=*+1   COMMAND AUXILIARY BYTE 1
030B          DAUX2   *=*+1
                ;
030C          TIMER1  *=*+2   INITIAL TIMER VALUE
030E          ADDCOR  *=*+1   ADDITION CORRECTION
030F          CASFLG  *=*+1   CASSETTE MODE WHEN SET
0310          TIMER2  *=*+2   FINAL TIMER VALUE. THESE TWO TIMER VALUES
                              ARE USED TO COMPUTE INTERVAL FOR BAUD RATE
0312          TEMP1   *=*+2   TEMPORARY STORAGE REGISTER
0314          TEMP2   *=*+1   TEMPORARY STORAGE REGISTER
0315          TEMP3   *=*+1   TEMPORARY STORAGE REGISTER
0316          SAVIO   *=*+1   SAVE SERIAL IN DATA PORT
0317          TIMFLG  *=*+1   TIME OUT FLAG FOR BAUD RATE CORRECTION
0318          STACKP  *=*+1   SIO STACK POINTER SAVE CELL
0319          TSTAT   *=*+1   TEMPORARY STATUS HOLDER
                ;
                ;
                ;
031A          HATABS  *=*+38  HANDLER ADDRESS TABLE
0021          MAXDEV  = *-HATABS-5    MAXIMUM HANDLER ADDRESS INDEX
                ;
                ;      NOTE : THE ENTIRE IOCB DEFINITIONS HAVE BEEN MODIFIED
0340          IOCB    *=*     I/O CONTROL BLOCKS
0340          ICHID   *=*+1   HANDLER INDEX NUMBER (FF = IOCB FREE)
0341          ICDNO   *=*+1   DEVICE NUMBER (DRIVE NUMBER)
0342          ICCOM   *=*+1   COMMAND CODE
0343          ICSTA   *=*+1   STATUS OF LAST IOCB ACTION
0344          ICBAL   *=*+1   BUFFER ADDRESS LOW BYTE
0345          ICBAH   *=*+1
0346          ICPTL   *=*+1   PUT BYTE ROUTINE ADDRESS - 1
0347          ICPTH   *=*+1
0348          ICBLL   *=*+1   BUFFER LENGTH LOW BYTE
0349          ICBLH   *=*+1
034A          ICAX1   *=*+1   AUXILIARY INFORMATION FIRST BYTE
034B          ICAX2   *=*+1
034C          ICSPR   *=*+4   FOUR SPARE BYTES
                      *=*+MAXIOC-IOCBSZ
03C0          PRNBUF  *=*+40  PRINTER BUFFER
                      *=*+21  SPARE BYTES
                ;
                ;
                ;
                ;
                ;
                ;      PAGE FOUR RAM ASSIGNMENTS
                ;
03FD          CASBUF  *=*+131 CASSETTE BUFFER
                ;
                ; USER AREA STARTS HERE AND GOES TO END OF PAGE SIX
                                                                FIVE
0480          USAREA  *=*+128 SPACE
                ;
                ;
                ;
                ;
                ;
                ;
                ;          PAGE FIVE RAM ASSIGNMENTS
                ;
                ;          PAGE FIVE IS RESERVED AS A USER WORK SPACE
                ;
                ;          NOTE: SEE FLOATING POINT SUBROUTINE AREA FOR PAGE FIVE CELLS
                ;
                ;
                ;          PAGE SIX RAM ASSIGNMENTS
                ;
                ;          PAGE SIX IS RESERVED AS A ~~USER~~ S USER WORK SPACE
                ;
                ;
                ;
                ;                        FLOATING POINT SUBROUTINES
                ;
0006            FPREC   =      6         ;FLOATING PT PRECISION (# OF BYTES)
                                         ;IF CARRY USED THEN CARRY CLEAR => NO ERROR, CARRY SET => ERROR
D800            AFP     =      $D800     ;ASCII->FLOATING POINT (FP)
                                         ;INBUFF+CIX -> FR0, CIX, CARRY
D8E6            FASC    =      $D8E6     ;FP -> ASCII        FR0 -> LBUFF  (INBUFF)
D9AA            IFP     =      $D9AA     ;INTEGER -> FP
                                         ;0-$FFFF (LSB,MSB) IN FR0,FR0+1->FR0
D9D2            FPI     =      $D9D2     ;FP -> INTEGER      FR0 -> FR0,FR0+1, CARRY
DA60            FSUB    =      $DA60     ;FR0 <- FR0 - FR1   ,CARRY
DA66            FADD    =      $DA66     ;FR0 <- FR0 + FR1   ,CARRY
DADB            FMUL    =      $DADB     ;FR0 <- FR0 * FR1   ,CARRY
DB28            FDIV    =      $DB28     ;FR0 <- FR0 / FR1   ,CARRY
DD89            FLD0R   =      $DD89     ;FLOATING LOAD REG0      FR0 <- (X,Y)
DD8D            FLD0P   =      $DD8D     ;   "      "     "       FR0 <- (FLPTR)
DD98            FLD1R   =      $DD98     ;   "      "    REG1     FR1 <- (X,Y)
DD9C            FLD1P   =      $DD9C     ;   "      "     "       FR1 <- (FLPTR)
DDA7            FST0R   =      $DDA7     ;FLOATING STORE REG0 (X,Y) <- FR0
DDAB            FST0P   =      $DDAB     ;   "      "     "   (FLPTR)<- FR0
DDB6            FMOVE   =      $DDB6     ;FR1 <- FR0
DD40            PLYEVL  =      $DD40     ;FR0 <- P(Z) = SUM(I=N TO 0) (A(I)*Z**I)      CARRY
                                         ;INPUT:  (X,Y) = A(N),A(N-1)...A(0)  -> PLYARG
                                         ;        ACC   = # OF COEFFICIENTS = DEGREE+1
                                         ;        FR0   = Z
                ;
                ;
DDC0            EXP     =      $DDC0     ;FR0 <- E**FR0 = EXP10(FR0 * LOG10(E))  CARRY
DDCC            EXP10   =      $DDCC     ;FR0 <- 10**FR0  CARRY
DECD            LOG     =      $DECD     ;FR0 <- LN(FR0) = LOG10(FR0)/LOG10(E)   CARRY
DED1            LOG10   =      $DED1     ;FR0 <- LOG10 (FR0)     CARRY
                ;
                ;                        ;THE FOLLOWING ARE IN BASIC CARTRIDGE:
BD81            SIN     =      $BD81     ;FR0 <- SIN(FR0)  DEGFLG=0 =>RADS, 6=>DEG     CARRY
BD73            COS     =      $BD73     ;FR0 <- COS(FR0)     CARRY
BE43            ATAN    =      $BE43     ;FR0 <- ATAN(FR0)    CARRY
BEB1            SQR     =      $BEB1     ;FR0 <- SQUAREROOT(FR0)    CARRY
                ;
                ;
                ;FLOATING POINT ROUTINES ZERO PAGE (NEEDED ONLY IF F.P. ROUTINES ARE CALLED)
                        *=$D4
00D4            FR0     *=*+FPREC        ;FP REG0
00DA            FRE     *=*+FPREC
00E0            FR1     *=*+FPREC        ;FP REG1
00E6            FR2     *=*+FPREC
00EC            FRX     *=*+1            ;SP BYTE
```

```
      00ED           EEXP    *=*+1                  ;VALUE OF E
      00EE           NSIGN   *=*+1                  ;SIGN OF #
      00EF           ESIGN   *=*+1                  ;SIGN OF EXPONENT
      00F0           FCHRFLG *=*+1                  ;1ST CHAR FLAG
      00F1           DIGRT   *=*+1                  ;# OF DIGITS RIGHT OF DECIMAL
      00F2           CIX     *=*+1                  ;CURRENT INPUT INDEX
      00F3           INBUFF  *=*+2                  ;POINTS TO USER'S LINE INPUT BUFFER
      00F5           ZTEMP1  *=*+2
      00F7           ZTEMP4  *=*+2
      00F9           ZTEMP3  *=*+2
      00FB           DEGFLG
      00FB           RADFLG  *=*+1                  ;0=RADIANS, 6=DEGREES
      0000           RADON   =        0             ;INDICATES RADIANS
      0006           DEGON   =        6             ;INDICATES DEGREES
      00FC           FLPTR   *=*+2                  ;POINTS TO USER'S FLOATING PT NUMBER
      00FE           FPTR2   *=*+2

                     ;FLOATING PT ROUTINES' NON-ZERO PAGE RAM (NEEDED ONLY IF F.P. ROUTINES CALLED)
                             *=$57E
      057E           LBPR1   *=*+1                  ;LBUFF PREFIX 1
      057F           LBPR2   *=*+1                  ;LBUFF PREFIX 2
      0580           LBUFF   *=*+128                ;LINE BUFFER
      05E0           PLYARG  =        LBUFF+$60     ;POLYNOMIAL ARGUMENTS
      05E6           FPSCR   =        PLYARG+FPREC
      05EC           FPSCR1  =        FPSCR+FPREC
      05E6           FSCR    =        FPSCR
      05EC           FSCR1   =        FPSCR1
      05FF           LBFEND  =        *-1           ;END OF LBUFF
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;        ;COLLEEN MNEMONICS
                     ;
      D200           POKEY   =        $D200         ;VBLANK ACTION:         DESCRIPTION:
      D200           POT0    =        POKEY+0       ;POT0-->PADDL0          0-227 IN RAM CELL
      D201           POT1    =        POKEY+1       ;POT1-->PADDL1          0-227 IN RAM CELL
      D202           POT2    =        POKEY+2       ;POT2-->PADDL2          0-227 IN RAM CELL
      D203           POT3    =        POKEY+3       ;POT3-->PADDL3          0-227 IN RAM CELL
      D204           POT4    =        POKEY+4       ;POT4-->PADDL4          0-227 IN RAM CELL
      D205           POT5    =        POKEY+5       ;POT5-->PADDL5          0-227 IN RAM CELL
      D206           POT6    =        POKEY+6       ;POT6-->PADDL6          0-227 IN RAM CELL
      D207           POT7    =        POKEY+7       ;POT7-->PADDL7          0-227 IN RAM CELL
      D208           ALLPOT  =        POKEY+8       ;???
      D209           KBCODE  =        POKEY+9
      D20A           RANDOM  =        POKEY+10
      D20B           POTGO   =        POKEY+11      ;STROBED
      D20D           SERIN   =        POKEY+13
      D20E           IRQST   =        POKEY+14
      D20F           SKSTAT  =        POKEY+15
      D200           AUDF1   =        POKEY+0
      D201           AUDC1   =        POKEY+1
      D202           AUDF2   =        POKEY+2
      D203           AUDC2   =        POKEY+3
      D204           AUDF3   =        POKEY+4
```

```
          D204              AUDF3    =      POKEY+4


   GENERAL INPUT/OUTPUT (GIO) 2-7-79

          D205              AUDC3    =      POKEY+5
          D206              AUDF4    =      POKEY+6
          D207              AUDC4    =      POKEY+7
          D208              AUDCTL   =      POKEY+8        NONE                      AUDCTL<--[$00]
          D209              STIMER   =      POKEY+9
          D20A              SKRES    =      POKEY+10       NONE                      SKRES<--[$00]
          D20D              SEROUT   =      POKEY+13       NONE                      SEROUT<--[$00]
          D20E              IRQEN    =      POKEY+14       POKMSK-->IRQEN   (AFFECTED BY OPEN S: OR E:)
          D20F              SKCTL    =      POKEY+15       SSKCTL-->SKCTL            SSKCTL<--[$13]

          D000              CTIA     =      $D000          VBLANK ACTION             DESCRIPTION
          D000              HPOSP0   =      CTIA+0
          D001              HPOSP1   =      CTIA+1
          D002              HPOSP2   =      CTIA+2
          D003              HPOSP3   =      CTIA+3
          D004              HPOSM0   =      CTIA+4
          D005              HPOSM1   =      CTIA+5
          D006              HPOSM2   =      CTIA+6
          D007              HPOSM3   =      CTIA+7
          D008              SIZEP0   =      CTIA+8
          D009              SIZEP1   =      CTIA+9
          D00A              SIZEP2   =      CTIA+10
          D00B              SIZEP3   =      CTIA+11
          D00C              SIZEM    =      CTIA+12
          D00D              GRAFP0   =      CTIA+13
          D00E              GRAFP1   =      CTIA+14
          D00F              GRAFP2   =      CTIA+15
          D010              GRAFP3   =      CTIA+16
          D011              GRAFM    =      CTIA+17
          D012              COLPM0   =      CTIA+18        PCOLR0-->COLPM0           WITH ATTRACT MODE
          D013              COLPM1   =      CTIA+19        PCOLR1-->COLPM1           WITH ATTRACT MODE
          D014              COLPM2   =      CTIA+20        PCOLR2-->COLPM2           WITH ATTRACT MODE
          D015              COLPM3   =      CTIA+21        PCOLR3-->COLPM3           WITH ATTRACT MODE
          D016              COLPF0   =      CTIA+22        COLOR0-->COLPF0           WITH ATTRACT MODE
          D017              COLPF1   =      CTIA+23        COLOR1-->COLPF1           WITH ATTRACT MODE
          D018              COLPF2   =      CTIA+24        COLOR2-->COLPF2           WITH ATTRACT MODE
          D019              COLPF3   =      CTIA+25        COLOR3-->COLPF3           WITH ATTRACT MODE
          D01A              COLBK    =      CTIA+26        COLOR4-->COLBK            WITH ATTRACT MODE
          D01B              PRIOR    =      CTIA+27        (ON OPEN S: OR E:)        GPRIOR-->PRIOR
          D01C              VDELAY   =      CTIA+28
          D01D              GRACTL   =      CTIA+29
          D01E              HITCLR   =      CTIA+30
          D01F              CONSOL   =      CTIA+31        #08-->CONSOL              TURN OFF SPEAKER
          D000              M0PF     =      CTIA+0
          D001              M1PF     =      CTIA+1
          D002              M2PF     =      CTIA+2
          D003              M3PF     =      CTIA+3
          D004              P0PF     =      CTIA+4
          D005              P1PF     =      CTIA+5
          D006              P2PF     =      CTIA+6
          D007              P3PF     =      CTIA+7
          D008              M0PL     =      CTIA+8
          D009              M1PL     =      CTIA+9
          D00A              M2PL     =      CTIA+10
          D00B              M3PL     =      CTIA+11
          D00C              P0PL     =      CTIA+12
          D00D              P1PL     =      CTIA+13
          D00E              P2PL     =      CTIA+14
          D00F              P3PL     =      CTIA+15
          D010              TRIG0    =      CTIA+16        TRIG0-->STRIG0
```

```
D011          TRIG1   =     CTIA+17        TRIG1-->STRIG1
D012          TRIG2   =     CTIA+18        TRIG2-->STRIG2
D013          TRIG3   =     CTIA+19        TRIG3-->STRIG3
              ;
D400          ANTIC   =     $D400          VBLANK ACTION          DESCRIPTION
D400          DMACTL  =     ANTIC+0        DMACTL<--SDMCTL        ON OPEN S: OR E:
D401          CHACTL  =     ANTIC+1        CHACTL<--CHACT         ON OPEN S: OR E:
D402          DLISTL  =     ANTIC+2        DLISTL<--SDLSTL        ON OPEN S: OR E:
D403          DLISTH  =     ANTIC+3        DLISTH<--SDLSTH        ON OPEN S: OR E:
D404          HSCROL  =     ANTIC+4
D405          VSCROL  =     ANTIC+5
D407          PMBASE  =     ANTIC+7
D409          CHBASE  =     ANTIC+9        CHBASE<--CHBAS         ON OPEN S: OR E:
D40A          WSYNC   =     ANTIC+10
D40B          VCOUNT  =     ANTIC+11
D40C          PENH    =     ANTIC+12
D40D          PENV    =     ANTIC+13
D40E          NMIEN   =     ANTIC+14       NMIEN<--40              POWER ON AND [SETVBV]
D40F          NMIRES  =     ANTIC+15       STROBED
D40F          NMIST   =     ANTIC+15
D300          PIA     =     $D300          VBLANK ACTION          DESCRIPTION
D300          PORTA   =     PIA+0          PORTA-->STICK0,1       X-Y CONTROLLERS
D301          PORTB   =     PIA+1          PORTB-->STICK2,3       X-Y CONTROLLERS
D302          PACTL   =     PIA+2          NONE                   PACTL<--3C [INIT]
D303          PBCTL   =     PIA+3          NONE                   PBCTL<--3C [INIT]
              ;
              ;
              .END
```

```
                              TITLE  "CENTRAL INPUT/OUTPUT (CIO) 2-7-79"
                      ;              UPDATED BY AL MILLER 3-9-79
     0030             ASCZER = '0           ASCII ZERO
     003A             COLON  = $3A          ASCII COLON
     009B             EOL    = $9B          END OF RECORD
```

```
                        ;
                        ; CIO JUMP VECTOR FOR USERS
                                *=CIOV
    E456 4C C4 E4               JMP     CIO             GO TO CIO
                        ;
                        ; CIO INIT JUMP VECTOR FOR POWER UP
                                *=CIOINV
    E46E 4C A6 E4               JMP     CIOINT          GO TO INIT
                        ;
                        ;
                        ; ERROR ROUTINE ADDRESS EQUATE
    00E4                ERRTNH=ERRTN/256
    00C0                ERRTNL=-ERRTNH*256+ERRTN
                        ;
                        ;
                                *=CIOORG
                        ;
                        ; CIO INITIALIZATION (CALLED BY MONITOR AT POWER UP)
    E4A6 A2 00          CIOINT  LDX     #0
    E4A8 A9 FF          CIOI1   LDA     #IOCFRE         SET ALL IOCB'S TO FREE
    E4AA 9D 40 03               STA     ICHID,X         BY SETTING HANDLER ID'S=$FF
    E4AD A9 C0                  LDA     #ERRTNL
    E4AF 9D 46 03               STA     ICPTL,X         POINT PUT TO ERROR ROUTINE
    E4B2 A9 E4                  LDA     #ERRTNH
    E4B4 9D 47 03               STA     ICPTH,X
    E4B7 8A                     TXA
    E4B8 18                     CLC
    E4B9 69 10                  ADC     #IOCBSZ         BUMP INDEX BY SIZE
    E4BB AA                     TAX
    E4BC C9 80                  CMP     #MAXIOC         DONE?
    E4BE 90 E8                  BCC     CIOI1           NO
    E4C0 60                     RTS                     YES, RETURN
                        ;
                        ; ERROR ROUTINE FOR ILLEGAL PUT
    E4C0                ERRTN=*-1
    E4C1 A0 85                  LDY     #NOTOPN         IOCB NOT OPEN
    E4C3 60                     RTS
```

```
                     ;
                     ; CIO LOCAL RAM (USES SPARE BYTES IN ZERO PAGE IOCB)
002E                 ENTVEC  =  ICSPRZ
                     ;
                     ; CIO MAIN ROUTINE
                     ;
                     ; CIO INTERFACES BETWEEN USER AND INPUT/OUTPUT DE
E4C4 85 2F           CIO     STA    CIOCHR          SAVE POSSIBLE OUTPUT CHARACTER
E4C6 86 2E                   STX    ICIDNO          SAVE IOCB NUMBER * N
                     ;
                     ; CHECK FOR LEGAL IOCB
E4C8 8A                      TXA
E4C9 29 0F                   AND    #$F             IS IOCB MULTIPLE OF 16?
E4CB D0 04                   BNE    CIERR1          NO, ERROR
E4CD E0 80                   CPX    #MAXIOC         IS INDEX TOO LARGE?
E4CF 90 05                   BCC    IOC1            NO
                     ;
                     ; INVALID IOCB NUMBER -- RETURN ERROR
E4D1 A0 86           CIERR1  LDY    #BADIOC         ERROR  CODE
E4D3 4C 1B E6                JMP    CIRTN1          RETURN
                     ;
                     ; MOVE USER IOCB TO ZERO PAGE
E4D6 A0 00           IOC1    LDY    #0
E4D8 BD 40 03        IOC1A   LDA    IOCB,X          USER IOCB
E4DB 99 20 00                STA    IOCBAS,Y        TO ZERO PAGE
E4DE E8                      INX
E4DF C8                      INY
E4E0 C0 0C                   CPY    #12             12 BYTES
E4E2 90 F4                   BCC    IOC1A
                     ;
                     ; COMPUTE CIO INTERNAL VECTOR FOR COMMAND
E4E4 A0 84                   LDY    #NVALID         ASSUME INVALID CODE
E4E6 A5 22                   LDA    ICCOMZ          COMMAND CODE TO INDEX
E4E8 C9 03                   CMP    #OPEN           IS COMMAND LEGAL?
E4EA 90 25                   BCC    CIERR4          NO -- <3.
E4EC A8                      TAY
                     ;
                     ; MOVE COMMAND TO ZERO BASE FOR INDEX
E4ED C0 0E                   CPY    #SPECIL         IS COMMAND SPECIAL?
E4EF 90 02                   BCC    IOC2            NO
E4F1 A0 0E                   LDY    #SPECIL         YES, SET SPECIAL OFFSET INDEX
E4F3 84 17           IOC2    STY    ICCOMT          SAVE COMMAND FOR VECTOR
E4F5 B9 C6 E6                LDA    COMTAB-3,Y      GET VECTOR OFFSET FROM TABLE
E4F8 F0 0E                   BEQ    CIOPEN          GO IF OPEN COMMAND
E4FA C9 02                   CMP    #2              IS IT CLOSE?
E4FC F0 35                   BEQ    CICLOS          YES
E4FE C9 08                   CMP    #8              IS IT STATUS OR SPECIAL?
E500 B0 4C                   BCS    CISTSP          YES
E502 C9 04                   CMP    #4              IS IT READ?
E504 F0 63                   BEQ    CIREAD          YES
E506 4C C9 E5                JMP    CIWRIT          ELSE, MUST BE WRITE
```

```
                        ;
                        ; OPEN COMMAND
                        ;
                        ; FIND DEVICE HANDLER IN HANDLER ADDRESS TABLE
E509 A5 20              CIOPEN  LDA     ICHIDZ          GET HANDLER ID
E50B C9 FF                      CMP     #IOCFRE         IS THIS IOCB CLOSED?
E50D F0 05                      BEQ     IOC6            YES
                        ;
                        ; ERROR -- IOCB ALREADY OPEN
E50F A0 81              CIERR3  LDY     #PRVOPN         ERROR CODE
E511 4C 1B E6           CIERR4  JMP     CIRTN1          RETURN
                        ;
                        ; GO FIND DEVICE
E514 20 9E E6           IOC6    JSR     DEVSRC          CALL DEVICE SEARCH
E517 B0 F8                      BCS     CIERR4          GO IF DEVICE NOT FOUND
                        ;
                        ; DEVICE FOUND, INITIALIZE IOCB FOR OPEN
                        ;
                        ; COMPUTE HANDLER ENTRY POINT
E519 20 3D E6           IOC7    JSR     COMENT
E51C B0 F3                      BCS     CIERR4          GO IF ERROR IN COMPUTE
                        ;
                        ; GO TO HANDLER FOR INITIALIZATION
E51E 20 B9 E6                   JSR     GOHAND          USE INDIRECT JUMP
                        ;
                        ; STORE PUT BYTE ADDRESS-1 INTO IOCB
E521 A9 0B                      LDA     #PUTCHR         SIMULATE PUT CHARACTER
E523 85 17                      STA     ICCOMT
E525 20 3D E6                   JSR     COMENT          COMPUTE ENTRY POINT
E528 A5 2C                      LDA     ICSPRZ          MOVE COMPUTED VALUE
E52A 85 26                      STA     ICPTLZ          TO PUT BYTE ADDRESS
E52C A5 2D                      LDA     ICSPRZ+1
E52E 85 27                      STA     ICPTHZ
E530 4C 1D E6                   JMP     CIRTN2          RETURN TO USER
```

```
                        ;
                        ;
                        ; CLOSE COMMAND
E533 A0 01              CICLOS  LDY     #SUCCES         ASSUME GOOD CLOSE
E535 84 23                      STY     ICSTAZ
E537 20 3D E6                   JSR     COMENT          COMPUTE HANDLER ENTRY POINT
E53A B0 03                      BCS     CICLO2          GO IF ERROR IN COMPUTE
E53C 20 89 E6                   JSR     GOHAND          GO TO HANDLER TO CLOSE DEVICE
E53F A9 FF              CICLO2  LDA     #IOCFRE         GET IOCB "FREE" VALUE
E541 85 20                      STA     ICHIDZ          SET HANDLER ID
E543 A9 E4                      LDA     #ERRTNH
E545 85 27                      STA     ICPTHZ          SET PUT BYTE TO POINT TO ERROR
E547 A9 C0                      LDA     #ERRTNL
E549 85 26                      STA     ICPTLZ
E54B 4C 1B E6                   JMP     CIRTN2          RETURN
                        ;
                        ;
                        ; STATUS AND SPECIAL REQUESTS
                        ; DO IMPLIED OPEN IF NECESSARY AND GO TO DEVICE
E54E A5 20              CISTSP  LDA     ICHIDZ          IS THERE A HANDLER ID?
E550 C9 FF                      CMP     #IOCFRE
E552 D0 05                      BNE     CIST1           YES
                        ;
                        ; IOCB IS FREE, DO IMPLIED OPEN
E554 20 9E E6                   JSR     DEVSRC          FIND DEVICE IN TABLE
E557 B0 08                      BCS     CIERR4          GO IF ERROR IN COMPUTE
                        ;
                        ; COMPUTE AND GO TO ENTRY POINT IN HANDLER
E559 20 3D E6           CIST1   JSR     COMENT          COMPUTER HANDLER ENTRY VECTOR
E55C 20 89 E6                   JSR     GOHAND          GO TO HANDLER
                        ;
                        ; RESTORE HANDLER INDEX (DO IMPLIED CLOSE)
E55F A6 2E                      LDX     ICIDNO          IOCB INDEX
E561 BD 40 03                   LDA     ICHID,X         GET ORIGINAL HANDLER ID
E564 85 20                      STA     ICHIDZ          RESTORE ZERO PAGE
E566 4C 1B E6                   JMP     CIRTN2          RETURN
```

```
                        ;
                        ; READ -- DO GET COMMAND
E569 A5 22              CIREAD  LDA     ICCOMZ          GET COMMAND BYTE
E56B 25 2A                      AND     ICAX1Z          IS THIS READ LEGAL?
E56D D0 05                      BNE     RCI1A           YES
                        ;
                        ; ILLEGAL READ -- IOCB OPENED FOR WRITE ONLY
E56F A0 83                      LDY     #WRONLY         ERROR CODE
E571 4C 1B E6           RCI1B   JMP     CIRTN1          RETURN
                        ;
                        ; COMPUTE AND CHECK ENTRY POINT
E574 20 3D E6           RCI1A   JSR     COMENT          COMPUTE ENTRY POINT
E577 B0 F8                      BCS     RCI1B           GO IF ERROR IN COMPUTE
                        ;
                        ; GET RECORD OR CHARACTERS
E579 A5 28                      LDA     ICBLLZ
E57B 05 29                      ORA     ICBLLZ+1        IS BUFFER LENGTH ZERO?
E57D D0 08                      BNE     RCI3            NO
E57F 20 89 E6                   JSR     GOHAND
E582 85 2F                      STA     CIOCHR
E584 4C 1D E6                   JMP     CIRTN2
                        ;
                        ; LOOP TO FILL BUFFER OR END RECORD
E587 20 89 E6           RCI3    JSR     GOHAND          GO TO HANDLER TO GET BYTE
E58A 85 2F                      STA     CIOCHR          SAVE BYTE
E58C 30 35                      BMI     RCI4            END TRANSFER IF ERROR
E58E A0 00                      LDY     #0
E590 91 24                      STA     (ICBALZ),Y      PUT BYTE IN USER BUFFER
E592 20 70 E6                   JSR     INCBFP          INCREMENT BUFFER POINTER
E595 A5 22                      LDA     ICCOMZ          GET COMMAND CODE
E597 29 02                      AND     #2              IS IT GET RECORD?
E599 D0 0C                      BNE     RCI1            NO
                        ;
                        ; CHECK FOR EOL ON TEXT RECORDS
E59B A5 2F                      LDA     CIOCHR          GET BYTE
E59D C9 9B                      CMP     #EOL            IS IT AN EOL?
E59F D0 06                      BNE     RCI1            NO
E5A1 20 63 E6                   JSR     DECBFL          YES, DECREMENT BUFFER LENGTH
E5A4 4C C3 E5                   JMP     RCI4            END TRANSFER
                        ;
                        ; CHECK BUFFER FULL
E5A7 20 63 E6           RCI1    JSR     DECBFL          DECREMENT BUFFER LENGTH
E5AA D0 DB                      BNE     RCI3            CONTINUE IF NON ZERO
```

[illegible]

```
                        ;
                        ; BUFFER FULL, RECORD NOT ENDED
                        ; DISCARD BYTES UNTIL END OF RECORD
E5AC A5 22              RCI2    LDA     ICCOMZ          GET COMMAND BYTE
E5AE 29 02                      AND     #2              IS IT GET CHARACTER?
E5B0 D0 11                      BNE     RCI4            YES, END TRANSFER
                        ;
                        ; LOOP TO WAIT FOR EOL
E5B2 20 89 E6           RCI6    JSR     GOHAND          GET BYTE FROM HANDLER
E5B5 85 2F                      STA     CIOCHR          SAVE CHARACTER
E5B7 30 0A                      BMI     RCI4            GO IF ERROR
                        ;
                        ; TEST RECORD, WAIT FOR EOL
E5B9 A5 2F                      LDA     CIOCHR          GET GOT BYTE
E5BB C9 9B                      CMP     #EOL            IS IT EOL?
E5BD D0 F3                      BNE     RCI6            NO, CONTINUE
                        ;
                        ; END OF RECORD, BUFFER FULL -- SEND TRUNCATED RECORD MESSAGE
E5BF A9 89              RCI11   LDA     #TRNRCD         ERROR CODE
E5C1 85 23                      STA     ICSTAZ          STORE IN IOCB
                        ;
                        ; TRANSFER DONE
E5C3 20 77 E6           RCI4    JSR     SUBBFL          SET FINAL BUFFER LENGTH
E5C6 4C 10 E6                   JMP     CIRTN2          RETURN
```

```
                        ;
                        ; WRITE -- DO PUT COMMANDS
E5C9 A5 22              CIWRIT  LDA     ICCOMZ          GET COMMAND BYTE
E5CB 25 2A                      AND     ICAX1Z          IS THIS WRITE LEGAL?
E5CD D0 05                      BNE     WCI1A           YES
                        ;
                        ; ILLEGAL WRITE -- DEVICE OPENED FOR READ ONLY
E5CF A0 87                      LDY     #RDONLY         ERROR CODE
E5D1 4C 1B E6           WCI1B   JMP     CIRTN1          RETURN
                        ;
                        ; COMPUTE AND CHECK ENTRY POINT
E5D4 20 3D E6           WCI1A   JSR     COMENT          COMPUTE HANDLER ENTRY POINT
E5D7 B0 F8                      BCS     WCI1B           GO IF ERROR IN COMPUTE
                        ;
                        ; PUT RECORD OR CHARACTERS
E5D9 A5 28                      LDA     ICBLLZ
E5DB 05 29                      ORA     ICBLLZ+1        IS BUFFER LENGTH ZERO?
E5DD D0 06                      BNE     WCI3            NO
E5DF A5 2F                      LDA     CIOCHR          GET CHARACTER
E5E1 E6 28                      INC     ICBLLZ          SET BUFFER LENGTH=1
E5E3 D0 06                      BNE     WCI4            THEN JUST TRANSFER ONE BYTE
                        ;
                        ; LOOP TO TRANSFER BYTES FROM BUFFER TO HANDLER
E5E5 A0 00              WCI3    LDY     #0
E5E7 B1 24                      LDA     (ICBALZ),Y      GET BYTE FROM BUFFER
E5E9 85 2F                      STA     CIOCHR          SAVE
E5EB 20 B9 E6           WCI4    JSR     GOHAND          GO PUT BYTE
E5EE 30 25                      BMI     WCI5            END IF ERROR
E5F0 20 70 E6                   JSR     INCBFP          INCREMENT BUFFER POINTER
                        ;
                        ; CHECK FOR TEXT RECORD
E5F3 A5 22                      LDA     ICCOMZ          GET COMMAND BYTE
E5F5 29 02                      AND     #2              IS IT PUT RECORD?
E5F7 D0 0C                      BNE     WCI1            NO
                        ;
                        ; TEXT RECORD -- CHECK FOR EOL TRANSFER
E5F9 A5 2F                      LDA     CIOCHR          GET LAST CHARACTER
E5FB C9 9B                      CMP     #EOL            IS IT AN EOL?
E5FD D0 06                      BNE     WCI1            NO
E5FF 20 63 E6                   JSR     DECBFL          DECREMENT BUFFER LENGTH
E602 4C 15 E6                   JMP     WCI5            END TRANSFER
                        ;
                        ; CHECK FOR BUFFER EMPTY
E605 20 63 E6           WCI1    JSR     DECBFL          DECREMENT BUFFER LENGTH
E608 D0 DB                      BNE     WCI3            CONTINUE IF NON ZERO
```

```
                        ;
                        ; BUFFER EMPTY, RECORD NOT FILLED
                        ; CHECK TYPE OF TRANSFER
E60A A5 22              WCI2    LDA     ICCOMZ          GET COMMAND CODE
E60C 29 02                      AND     #2              IS IT PUT CHARACTER?
E60E D0 05                      BNE     WCI5            YES, END TRANSFER
                        ;
                        ; PUT RECORD (TEXT), BUFFER EMPTY, SEND EOL
E610 A9 9B                      LDA     #EOL
E612 20 89 E6                   JSR     GOHAND          GO TO HANDLER
                        ;
                        ; END PUT TRANSFER
E615 20 73 E6           WCI5    JSR     SUBBFL          SET ACTUAL PUT BUFFER LENGTH
E618 4C 1D E6                   JMP     CIRTN2          RETURN
```

```
                           ;
                           ; CIO RETURNS
                           ; RETURNS WITH Y=STATUS
      E61B 84 23           CIRTN1  STY     ICSTAZ              SAVE STATUS
                           ;
                           ; RETURNS WITH STATUS STORED IN ICSTAZ
                           ; MOVE IOCB IN ZERO PAGE BACK TO USER AREA
      E61D A4 2E           CIRTN2  LDY     ICIDNO              GET IOCB INDEX
      E61F B9 44 03                LDA     ICBAL,Y
      E622 85 24                   STA     ICBALZ              RESTORE USER BUFFER POINTER
      E624 B9 45 03                LDA     ICBAH,Y
      E627 85 25                   STA     ICBAHZ
      E629 A2 00                   LDX     #0                  LOOP COUNT AND INDEX
      E62B B5 20           CIRT3   LDA     IOCBAS,X            ZERO PAGE
      E62D 99 40 03                STA     IOCB,Y              TO USER AREA
      E630 E8                      INX
      E631 C8                      INY
      E632 E0 0C                   CPX     #12                 12 BYTES
      E634 90 F5                   BCC     CIRT3
                           ;
                           ; RESTORE A,X, & Y
      E636 A5 2F                   LDA     CIOCHR              GET LAST CHARACTER
      E638 A6 2E                   LDX     ICIDNO              IOCB INDEX
      E63A A4 23                   LDY     ICSTAZ              GET STATUS AND SET FLAGS
      E63C 60                      RTS                         RETURN TO USER
```

```
                        ;
                        ; CIO SUBROUTINES
                        ;
                        ; COMENT -- CHECK AND COMPUTE HANDLER ENTRY POINT
E63D A4 20              COMENT  LDY     ICHIDZ          GET HANDLER INDEX
E63F C0 22                      CPY     #MAXDEV+1       IS IT A LEGAL INDEX?
E641 90 04                      BCC     COM1            YES
                        ;
                        ; ILLEGAL HANDLER INDEX MEANS DEVICE NOT OPEN FOR OPERATION
E643 A0 85                      LDY     #NOTOPN         ERROR CODE
E645 B0 1B                      BCS     COM2            RETURN
                        ;
                        ; USE HANDLER ADDRESS TABLE AND COMMAND TABLE TO GET VECTOR
E647 B9 1B 03           COM1    LDA     HATABS+1,Y      GET LOW BYTE OF ADDRESS
E64A 85 2C                      STA     ICSPRZ          AND SAVE IN POINTER
E64C B9 1C 03                   LDA     HATABS+2,Y      GET HI BYTE OF ADDRESS
E64F 85 2D                      STA     ICSPRZ+1
E651 A4 22                      LDY     ICCOMZ          GET COMMAND CODE
E653 B9 C6 E6                   LDA     COMTAB-3,Y      GET COMMAND OFFSET
E656 A8                         TAY
E657 B1 2C                      LDA     (ICSPRZ),Y      GET LOW BYTE OF VECTOR FROM
E659 AA                         TAX                     HANDLER ITSELF AND SAVE
E65A C8                         INY
E65B B1 2C                      LDA     (ICSPRZ),Y      GET HI BYTE OF VECTOR
E65D 85 2D                      STA     ICSPRZ+1
E65F 86 2C                      STX     ICSPRZ          SET LO BYTE
E661 18                         CLC                     SHOW NO ERROR
E662 60                 COM2    RTS
                        ;
                        ;
                        ; DECBFL -- DECREMENT BUFFER LENGTH DOUBLE BYTE
                        ; Z FLAG = 0 ON RETURN IF LENGTH = 0 AFTER DECREMENT
E663 C6 28              DECBFL  DEC     ICBLLZ          DECREMENT LOW BYTE
E665 A5 28                      LDA     ICBLLZ          CHECK IT
E667 C9 FF                      CMP     #$FF            DID IT GO BELOW?
E669 D0 02                      BNE     DECBF1          NO
E66B C6 29                      DEC     ICBLLZ+1        DECREMENT HI BYTE
E66D 05 29              DECBF1  ORA     ICBLLZ+1        SET Z IF BOTH ARE ZERO
E66F 60                         RTS
                        ;
                        ;
                        ; INCBFP -- INCREMENT WORKING BUFFER POINTER
E670 E6 24              INCBFP  INC     ICBALZ          BUMP LOW BYTE
E672 D0 02                      BNE     INCBF1          GO IF NOT ZERO
E674 E6 25                      INC     ICBALZ+1        ELSE, BUMP HI BYTE
E676 60                 INCBF1  RTS
                        ;
                        ;
                        ; SUBBFL -- SET BUFFER LENGTH = BUFFER LENGTH - WORKING BYTE COUNT
E677 A6 2E              SUBBFL  LDX     ICIDNO          GET IOCB INDEX
E679 38                         SEC
E67A BD 48 03                   LDA     ICBLL,X         GET LOW BYTE OF INITIAL LENGTH
E67D E5 28                      SBC     ICBLLZ          SUBTRACT FINAL LOW BYTE
E67F 85 28                      STA     ICBLLZ          AND SAVE BACK
E681 BD 49 03                   LDA     ICBLH,X         GET HI BYTE
E684 E5 29                      SBC     ICBLLZ+1
E686 85 29                      STA     ICBLHZ
E688 60                         RTS
```

```
                            ;
                            ; COMAND -- GO INDIRECT TO A DEVICE HANDLER
                            ; Y= STATUS ON RETURN, N FLAG=1 IF ERROR ON RETURN
E689 A0 92                  COMAND  LDY     #FNCNOT         PREPARE NO FUNCTION STATUS FOR HANDLER RTS
E68B 20 93 E6                       JSR     CIJUMP          USE THE INDIRECT JUMP
E68E 84 23                          STY     ICSTAZ          SAVE STATUS
E690 C0 00                          CPY     #0              AND SET N FLAG
E692 60                             RTS
                            ;
                            ; INDIRECT JUMP TO HANDLER BY PAUL'S METHOD
E693 AA                     CIJUMP  TAX                     SAVE A
E694 A5 2D                          LDA     ICSPRZ+1        GET JUMP ADDRESS HI BYTE
E696 48                             PHA                     PUT ON STACK
E697 A5 2C                          LDA     ICSPRZ          GET JUMP ADDRESS LO BYTE
E699 48                             PHA                     PUT ON STACK
E69A 8A                             TXA                     RESTORE A
E69B A6 2E                          LDX     ICIDNO          GET IOCB INDEX
E69D 60                             RTS                     GO TO HANDLER INDIRECTLY
```

```
                           ;
                           ; DEVSRC -- DEVICE SEARCH, FIND DEVICE IN HANDLER ADDRESS TABLE
                           ;
                           ; LOOP TO FIND DEVICE
     E69E A0 00            DEVSRC  LDY    #0
     E6A0 B1 24                    LDA    (ICBALZ),Y       GET DEVICE NAME FROM USER
     E6A2 F0 0C                    BEQ    CIERR2
     E6A4 A0 21                    LDY    #MAXDEV          INITIAL COMPARE INDEX
     E6A6 D9 1A 03         DEVS1   CMP    HATABS,Y         IS THIS THE DEVICE?
     E6A9 F0 0A                    BEQ    DEVS2            YES
     E6AB 88                       DEY
     E6AC 88                       DEY                     ELSE, POINT TO NEXT DEVICE NAME
     E6AD 88                       DEY
     E6AE 10 F6                    BPL    DEVS1            CONTINUE FOR ALL DEVICES
                           ;
                           ; NO DEVICE FOUND, DECLARE NON-EXISTENT DEVICE ERROR
     E6B0 A0 82            CIERR2  LDY    #NONDEV          ERROR CODE
     E6B2 38                       SEC                     SHOW ERROR
     E6B3 B0 13                    BCS    DEVS4            AND RETURN
                           ;
                           ; FOUND DEVICE, SET ICHID,ICDNO, AND INIT DEVICE
     E6B5 98               DEVS2   TYA
     E6B6 85 20                    STA    ICHIDZ           SAVE HANDLER INDEX
     E6B8 38                       SEC
     E6B9 A0 01                    LDY    #1
     E6BB B1 24                    LDA    (ICBALZ),Y       GET DEVICE NUMBER (DRIVE NUMBER)
     E6BD E9 30                    SBC    #ASCZER          SUBTRACT ASCII ZERO
     E6BF C9 0A                    CMP    #$A              IS NUMBER IN RANGE?
     E6C1 90 02                    BCC    DEVS3            YES
     E6C3 A9 01                    LDA    #1               NO, DEFAULT TO ONE
     E6C5 85 21            DEVS3   STA    ICDNOZ           SAVE DEVICE NUMBER
     E6C7 18                       CLC                     SHOW NO ERROR
                           ;
                           ; RETURN
     E6C8 60               DEVS4   RTS
```

[Bxx

better test.

```
                          ;
                          ;
                          ; CIO ROM TABLES
                          ;
                          ; COMMAND TABLE
                          ; MAPS EACH COMMAND TO OFFSET FOR APPROPRIATE VECTOR IN HANDLER
E6C9 00 04 04             COMTAB  .BYTE  0,4,4,4,4,6,6,6,6,2,8,10
E6CC 04 04 06
E6CF 06 06 06             ; command =    3 4 5 6 7 8 9 A B C D >D
E6D2 02 08 0A
022F                      LENGTH=*-CIOINT
E6D5                      CRNTPC  =*
                                  *=$14
0014 00                   CIOSPR  .BYTE  INTORG-CRNTPC   ;CIOL IS TOO LONG
                                  .END
```

```
CENTRAL INPUT/OUTPUT (CIO) 2-7-79

              SYMBOL TABLE
     ADDCOR  0062    ADRESS  0064    AFP     D800    ALLPOT  D20B
     ANTIC   D400    APPEND  0001    APPMHI  000E    ASCZER  0030
     ATACHR  02FB    ATAN    BE43    ATRACT  004D    AUDC1   D201
     AUDC2   D203    AUDC3   D205    AUDC4   D207    AUDCTL  D208
     AUDF1   D200    AUDF2   D202    AUDF3   D204    AUDF4   D206
     BADTOE  0095    BADMOD  0097    BFENHI  0035    BFENLO  0034
     BITMSK  006E    BLIM    028A    BLKBDV  E471    BOOT?   0009
     BOOTAD  0242    BOTSCR  02BF    BPTR    003D    BRKABT  0080
     BRKKEY  0011    BUFADR  0015    BUFCNT  006B    BUFRFL  0038
     BUFRHI  0033    BUFRLO  0032    BUFSTR  006C    CASBUF  03FD
     CASETV  E440    CASFLG  030F    CASINI  0002    CASORG  EF41
     CASSBT  0040    CASSET  0043    CAUX1   023C    CAUX2   023D
     CBAUDH  02EF    CBAUDL  02EE    CCOMND  023B    CDEVIC  023A
     CDTMA1  0226    CDTMA2  0228    CDTMF3  022A    CDTMF4  022C
     CDTMF5  022E    CDTMV1  0218    CDTMV2  021A    CDTMV3  021C
     CDTMV4  021E    CDTMV5  0220    CH      02FC    CH1     02F2
     CHACT   02F3    CHACTL  D401    CHAR    02FA    CHBAS   02F4
     CHBASE  D409    CHKERR  008F    CHKSNT  003B    CHKSUM  0031
     CIBORG  E000    CICLO2  E53F    CICLOS  E533    CIERR1  E4D1
     CIERR2  E6D0    CIERR3  E50F    CIERR4  E511    CIJUMP  E693
     CIO     E4C4    CIOCHR  002F    CIOI1   E4A8    CIOINT  E4A6
     CIOINV  E46E    CIOORG  E4A6    CIOPEN  E509    CIOSPR  0014
     CIOV    E456    CIREAD  E567    CIRTN   E62B    CIRTN1  E61B
     CIRTN2  E61D    CIST1   E559    CISTSP  E54E    CIWRIT  E5C9
     CIX     00E2    CKEY    004A    CLOSE   000C    COLAC   0072
     COLBK   D01A    COLCRS  0055    COLDST  0244    COLDSV  E477
     COLINC  007A    COLON   003A    COLOR0  02C4    COLOR1  02C5
     COLOR2  02C6    COLOR3  02C7    COLOR4  02C8    COLPF0  D016
     COLPF1  D017    COLPF2  D018    COLPF3  D019    COLPM0  D012
     COLPM1  D013    COLPM2  D014    COLPM3  D015    COLRSH  004F
     COM1    E6A7    COM2    E662    COMENT  E63D    COMTAB  E6C9
     CONSOL  D01F    COS     BD73    COUNTR  007E    CR      009B
     CRETRY  0036    CRITIC  0042    CRNTP2  E6D5    CRSINH  02F0
     CRSROW  008D    CSOPIV  E47D    CSTAT   0288    CTIA    D000
     DAUX1   030A    DAUX2   030B    DBSECT  0241    DBUFHI  0305
     DBUFLO  0304    DBYTHI  0309    DBYTLO  0308    DCB     0300
     DCOMND  0302    DDEVIC  0300    DECBFI  E66D    DECBFL  E663
     DEGFLG  00FB    DEGON   0006    DELETE  0021    DELTAC  0077
     DELTAR  0076    DERROR  0090    DEVSB1  E6A6    DEVSB2  E6B5
     DEVSB3  E6C5    DEVSB4  E6C9    DEVSBC  E69E    DFLAGS  0240
     DIGRT   00F1    DINDEX  0057    DIRECT  0002    DISK    0044
     DISKIV  E450    DISPLY  0053    DLISTH  D403    DLISTL  D402
     DMACTL  D400    DMASK   02A0    DNACK   008B    DOSINI  000C
     DOSVEC  000A    DRAWLN  0011    DRETRY  0037    DRKMSK  004E
     DSKFMS  0018    DSKINV  E453    DSKRDR  E0EA    DSKTIM  0246
     DSKUTL  001A    DSPFLG  02FE    DSTAT   004C    DSTATS  0303
     DTIMLO  0306    DUNIT   0301    DUNUSE  0307    DVSTAT  02EA
     EDITRV  E400    EEXP    00ED    ENDPT   0074    ENTVEC  002C
     EOFERR  0088    EOL     009B    ERRFLG  023F    ERRTN   E4C0
     ERRTN1  00F4    ERRTNL  00C0    ESCFLG  02A2    ESIGN   00EF
     EXP     DDC0    EXP10   DDCC    FADD    DA66    FASC    D8E6
     FCHRFL  00F0    FDIV    DB28    FEOF    003F    FILDAT  02FD
     FILFLG  02B7    FILLIN  0012    FLD0P   0080    FLD0R   DD89
     FLD1P   0098    FLD1R   DD98    FLPTR   00FC    FMOVE   DDB6
     FMSZPG  0043    FMUL    DADB    FNCNOT  0092    FNORMAT 00D2
     FP1     D2D0    FPREC   0006    FPSCR   05E6    FPSCR1  05EC
     FPTR2   00FE    FR0     00D4    FR1     00E0    FR2     00E6
     FRE     00DA    FREQ    0040    FRMADR  008C    FRX     00EC
     FSCR    05E6    FSCR1   05EC    FTOFP   D9AB    FSTOR   DDA7
```

CENTRAL INPUT/OUTPUT (CIO) 2-7-79

```
[illegible] [illegible]    FTYPE   [illegible]   GETCHR  0007    GETREC  0005
GLBABS  02E0    [illegible] [illegible]   GPRIOR  026F    GRACTL  D01D
GRAFM   D011    GRAFP0  D00D    GRAFP1  D00E    GRAFP2  D00F
GRAFP3  D010    HATABS  031A    HITCLR  D01E    HOLD1   0051
HOLD2   029F    HOLD3   029D    HOLD4   02BC    HOLD5   02BD
HOLDCH  007C    HPOSM0  D004    HPOSM1  D005    HPOSM2  D006
HPOSM3  D007    HPOSP0  D000    HPOSP1  D001    HPOSP2  D002
HPOSP3  D003    HSCROL  D404    ICAX1   034A    ICAX1Z  002A
ICAX2   034B    ICAX2Z  002B    ICBAH   0345    ICBAHZ  0025
ICBAL   0344    ICBALZ  0024    ICBLH   0349    ICBLHZ  0029
ICBLL   0348    ICBLLZ  0028    ICCOM   0342    ICCOMT  0017
ICCOMZ  0022    ICDNO   0341    ICDNOZ  0021    ICHID   0340
ICHIDZ  0020    ICIDNO  002E    ICPTH   0347    ICPTHZ  0027
ICPTL   0346    ICPTLZ  0026    ICSPR   034C    ICSPRZ  002C
ICSTA   0343    ICSTAZ  0023    IFP     D9AA    INBUFF  00F3
INCRF1  E676    INCRFP  E670    INSCLR  0020    INSDAT  007D
INTABS  0200    INTEMP  022D    INTINV  E46B    INTORG  E6D5
INTZBS  0010    INVFLG  02B6    IOC1    E4D6    IOC1A   E4DB
IOC2    E4E1    IOC6    E514    IOC7    E519    IOCB    0340
IOCBAS  0020    IOCBSZ  0010    IOCFRE  00FF    IRQEN   D20E
IRQST   D20E    KBCODE  D209    KBD     004B    KBDORG  F3E4
KEYBDV  E420    KEYDEL  02F1    LBFEND  00FF    LBPR1   057E
LBPR2   057F    LBUFF   0580    LEDGE   0002    LENGTH  022F
LINBUF  0247    LINZBS  0000    LMARGN  0052    LOCKFL  0023
LOG     DECD    LOG10   DED1    LOGCOL  0063    LOGMAP  02B2
LPENH   0234    LPENV   0235    M0PF    D000    M0PL    D008
M1PF    D001    M1PL    D009    M2PF    D002    M2PL    D00A
M3PF    D003    M3PL    D00B    MAXDEV  0021    MAXIOC  0080
MEMLO   02E7    MEMTOP  02E5    MLTTMP  0066    MODEM   004D
[illegible] F0F3    MXDMOD  0010    NEWCOL  0061    NEWROW  0060
NMIEN   D40E    NMIRES  D40F    NMIST   D40F    NOCKSM  003C
NONDEV  0082    NOTE    0026    NOTOPN  0085    NSIGN   00EE
NVALID  0084    OLDADR  005E    OLDCHR  005D    OLDCOL  005B
OLDROW  005A    OPEN    0003    OPNIN   0004    OPNINO  000C
OPNOT   0008    OPNTMP  0066    OVRRUN  008E    P0PF    D004
P0PL    D00C    P1PF    D005    P1PL    D00D    P2PF    D006
P2PL    D00E    P3PF    D007    P3PL    D00F    PACTL   D302
PADDL0  0270    PADDL1  0271    PADDL2  0272    PADDL3  0273
PADDL4  0274    PADDL5  0275    PADDL6  0276    PADDL7  0277
PBCTL   D303    PBPNT   001D    PBUFSZ  001E    PCOLR0  02C0
PCOLR1  02C1    PCOLR2  02C2    PCOLR3  02C3    PENH    D40C
PENV    D40D    PIA     D300    PLYARG  05E0    PLYEVL  D840
[illegible] D9D7    POINT   0035    POKEY   D200    POKMSK  0010
PORTA   D300    PORTB   D301    POT0    D200    POT1    D201
POT2    D202    POT3    D203    POT4    D204    POT5    D205
POT6    D206    POT7    D207    POTGO   D20B    PRINTR  0050
PRINTV  E430    PRIOR   D01B    PRNBUF  03C0    PRNORG  EE78
PRVOPN  0031    PTEMP   [illegible]    PTIMOT  001C    PTRIG0  027C
PTRIG1  027D    PTRIG2  027E    PTRIG3  027F    PTRIG4  0280
PTRIG5  0281    PTRIG6  0282    PTRIG7  0283    PUTCHR  000B
PUTREC  0009    RADFLG  00FB    RADON   0000    RAMLO   0004
RAMSIZ  02E4    RAMTOP  006A    RANDOM  D20A    RBLOKV  E47A
[illegible] [illegible]    [illegible] [illegible]    [illegible] E374    [illegible] E371
[illegible] [illegible]    [illegible] [illegible]    [illegible] [illegible]    [illegible] [illegible]
[illegible] [illegible]    RECVDN  [illegible]    [illegible] [illegible]    [illegible] [illegible]
[illegible] [illegible]    [illegible] [illegible]    [illegible] 0054    ROWINC  0079
RTCLOK  0012    SAVADR  0068    SAVIO   0316    SAVMSC  0058
[illegible] [illegible]    SCRENV  E410    [illegible] [illegible]    [illegible] 0093
[illegible] [illegible]    SDLSTL  0230    SDMCTL  022F    [illegible] [illegible]
[illegible] [illegible]    [illegible] [illegible]    [illegible] [illegible]    [illegible] [illegible]
```

```
CENTRAL INPUT/OUTPUT (CIO) 2-7-79

SHFLOK  02BE     SIN     BDB1     SIOINV  E465     SIOORG  E944
SIOV    E459     SIZEM   D00C     SIZEP0  D008     SIZEP1  D009
SIZEP2  D00A     SIZEP3  D00B     SKCTL   D20F     SKRES   D20A
SKSTAT  D20F     SOUNDR  0041     SPECTL  000E     SQR     BEB1
SKTIMR  022B     SSFLAG  02FF     SSKCTL  0232     STACKP  0318
STATIS  000B     STATUS  0030     STICK0  0278     STICK1  0277
STICK2  027A     STICK3  027B     STIMER  D209     STRIG0  0284
STRIG1  0285     STRIG2  0286     STRIG3  0287     SUBBFL  E677
SUBTMP  029E     SUCCES  0001     SWPFLG  007B     SYSVBV  E45F
TABMAP  02A3     TEMP    023E     TEMP1   0312     TEMP2   0314
TEMP3   0315     TIMER1  030C     TIMER2  0310     TIMFLG  0317
TINOUT  008A     TINDEX  0293     TMPCHR  0050     TMPCOL  02B9
TMPLBT  02A1     TMPROW  02B8     TMPX1   029C     TRAMSZ  0006
TRIG0   D010     TRIG1   D011     TRIG2   D012     TRIG3   D013
TRNRCD  0009     TSTAT   0319     TSTDAT  0007     TXTCOL  0291
TXTMSC  0294     TXTOLD  0296     TXTROW  0290     UNLOCK  0024
USAREA  0480     VBREAK  0206     VCOUNT  D40B     VCTABL  E480
VDELAY  D01C     VDSLST  0200     VECTBL  E400     VIMIRQ  0216
VINTER  0204     VKEYBD  0208     VPRCED  0202     VSCROL  D405
VSERIN  020A     VSEROC  020E     VSEROR  020C     VTIMR1  0210
VTIMR2  0212     VTIMR4  0214     VVBLKD  0224     VVBLKI  0222
WARMST  0008     WARMSV  E474     WCI1    E605     WCI1A   E5D4
WCI1B   E5D1     WCI2    E60A     WCI3    E5E5     WCI4    E5EB
WCI5    E615     WMODE   0289     WRONLY  0083     WSYNC   D40A
XITVBV  E462     XMTDON  003A     ZIOCB   0020     ZTEMP1  00F5
ZTEMP3  00F9     ZTEMP4  00F7
```

INTERRUPT HANDLER

```
                              .TITLE  'INTERRUPT HANDLER'
                        ;LIVES ON DK1:INTHV.SRC
 0006                   SRTIM2 =       6         SECOND REPEAT INTERVAL
                              *=SETVBV
 E45C 4C 12 E9                JMP     SETVBL
 E45F 4C D1 E7                JMP     SYSVBL
 E462 4C 3F E9                JMP     XITVBL
                              *=INTINV
 E46B 4C D5 E6                JMP     IHINIT
                        ;
                              * = VCTABL+INTABS-VDSLST
                        ;
 E480 B3 E7✓                  .WORD   SYRTI   VDSLST
 E482 B2 E7✓                  .WORD   SYIRQB  VPRCED
 E484 B2 E7✓                  .WORD   SYIRQB  VINTER
 E486 B2 E7✓                  .WORD   SYIRQB  VBREAK
                        ;
                              * = *+8
 E490 B2 E7✓                  .WORD   SYIRQB  VTIMR1
 E492 B2 E7✓                  .WORD   SYIRQB  VTIMR2
 E494 B2 E7✓                  .WORD   SYIRQB  VTIMR4
 E496 F6 E6✓                  .WORD   SYIRQ   VIMIRQ
 E498 00 00 00                .WORD   0,0,0,0,0        CDTMV1-4
 E49B 00 00 00
 E49E 00 00 00
 E4A1 00
 E4A2 D1 E7                   .WORD   SYSVBL  VVBLKI
 E4A4 3F E9                   .WORD   XITVBL  VVBLKD
                        ;
                              *=$900C
                        ;
 900C A9 E6                   LDA     #PIRQH           SET UP RAM VECTORS FOR LINBUG VERSION
 900E 8D F9 FF                STA     $FFF9
 9011 A9 F3                   LDA     #PIRQL
 9013 8D F8 FF                STA     $FFF8
 9016 A9 E7                   LDA     #PNMIH
 9018 8D FB FF                STA     $FFFB
 901B A9 B4                   LDA     #PNMIL
 901D 8D FA FF                STA     $FFFA
 9020 60                      RTS
```

```
                             ;
                             ; IRQ HANDLER
                             ;
                             ; JUMP THRU IMMEDIATE IRQ VECTOR, WHICH ORDINARILY POINTS TO
                             ; SYSTEM IRQ. DETERMINE & CLEAR CAUSE, JUMP THRU SOFTWARE VECTOR
                             ;
                                   *=INTORG
E6D5 A9 40           SETNMI  LDA   #$40     VBL ON BUT DLIST OFF***FOR NOW***
E6D7 8D 0E D4                STA   NMIEN    ENABLE DISPLAY LIST, VERTICAL BLANK
E6DA A9 38                   LDA   #$38     LOOK AT DATA DIRECTION REGISTERS IN PIA
E6DC 8D 02 D3                STA   PACTL
E6DF 8D 03 D3                STA   PBCTL
E6E2 A9 00                   LDA   #0       MAKE ALL INPUTS
E6E4 8D 00 D3                STA   PORTA
E6E7 8D 01 D3                STA   PORTB
E6EA A9 3C                   LDA   #$3C     BACK TO PORTS
E6EC 8D 02 D3                STA   PACTL
E6EF 8D 03 D3                STA   PBCTL
E6F2 60                      RTS
E6F3 6C 16 02        PIRQ    JMP   (VIMIRQ)
E6F6 48              SYIRQ   PHA            SAVE A ON STACK
E6F7 A9 10                   LDA   #$10     SERIAL OUT READ BIT
E6F9 2C 0E D2                BIT   IRQST    SEE IF SER OUT RDY
E6FC D0 0D                   BNE   SYIRQ1   JUMP IF NOT
E6FE A9 EF                   LDA   #$EF
E700 8D 0E D2                STA   IRQEN
E703 A5 10                   LDA   POKMSK   RESET ENABLE BYTE
E705 8D 0E D2                STA   IRQEN
E708 6C 0C 02                JMP   (VSEROR)  JUMP THRU VECTOR
E70B A9 20           SYIRQ1  LDA   #$20     SERIAL IN READY BIT
E70D 2C 0E D2                BIT   IRQST
E710 D0 0D                   BNE   SYIRQ2
E712 A9 DF                   LDA   #$DF
E714 8D 0E D2                STA   IRQEN
E717 A5 10                   LDA   POKMSK
E719 8D 0E D2                STA   IRQEN
E71C 6C 0A 02                JMP   (VSERIN)
E71F A9 08           SYIRQ2  LDA   #$08     SERIAL OUT COMPLETE
E721 24 10                   BIT   POKMSK   ONLY CHECK IF ENABLED
E723 F0 12                   BEQ   SYIRQ3   SINCE THIS IS A WIERD CASE IN THE
E725 2C 0E D2                BIT   IRQEN    HARDWARE
E728 D0 0D                   BNE   SYIRQ3
E72A A9 F7                   LDA   #$F7
E72C 8D 0E D2                STA   IRQEN
E72F A5 10                   LDA   POKMSK
E731 8D 0E D2                STA   IRQEN
E734 6C 0E 02                JMP   (VSEROC)
E737 AD 0E D2        SYIRQ3  LDA   IRQEN    TO CHECK TIMERS
E73A 6A                      ROR   A
E73B B0 0D                   BCS   SYIRQ4   TIMER1
E73D A9 FE                   LDA   #$FE
E73F 8D 0E D2                STA   IRQEN
E742 A5 10                   LDA   POKMSK
E744 8D 0E D2                STA   IRQEN
E747 6C 10 02                JMP   (VTIMR1)
E74A 6A              SYIRQ4  ROR   A
E74B B0 0D                   BCS   SYIRQ5
E74D A9 FD                   LDA   #$FD
E74F 8D 0E D2                STA   IRQEN
E752 A5 10                   LDA   POKMSK
```

INTERRUPT HANDLER

```
E754 8D 0E D2            STA    IRQEN
E757 6C 10 02            JMP    (VTIMR2)
E75A 0A          SYIRQ5  ASL    A
E75B 90 0A               BCC    SYIRQ6
E75D A9 FB               LDA    #$FB
E75F 8D 0E D2            STA    IRQEN
E762 A5 10               LDA    POKMSK
E764 8D 0E D2            STA    IRQEN                                   ?
                         JMP    (VTIMR4)
E767 2C 0E D2    SYIRQ6  BIT    IRQEN   CHECK BREAK KEY, KEYBOARD INTERRUPTS
E76A 70 0D               BVS    SYIRQ7  IF KEYBOARD
E76C A9 BF               LDA    #$BF
E76E 8D 0E D2            STA    IRQEN
E771 A5 10               LDA    POKMSK
E773 8D 0E D2            STA    IRQEN
E776 6C 08 02            JMP    (VKEYBD)
E779 30 18       SYIRQ7  BMI    SYIRQ8  BREAK KEY
E77B A9 7F               LDA    #$7F
E77D 8D 0E D2            STA    IRQEN
E780 A5 10               LDA    POKMSK
E782 8D 0E D2            STA    IRQEN
E785 A9 00               LDA    #0
E787 85 11               STA    BRKKEY  SET BREAK KEY FLAG
E789 8D FF 02            STA    SSFLAG  START/STOP FLAG
E78C 8D F0 02            STA    CRSINH  CURSOR INHIBIT
E78F 85 4D               STA    ATRACT  TURN OFF ATRACT MODE
E791 68                  PLA
E792 40                  RTI            EXIT FROM INT
E793 2C 02 D3    SYIRQ8  BIT    PACTL   PROCEED ***I GUESS***
E796 10 06               BPL    SYIRQ9
E798 AD 00 D3            LDA    PORTA   CLEAR INT STATUS BIT
E79B 6C 02 02            JMP    (VPRCED)
E79E 2C 03 D3    SYIRQ9  BIT    PBCTL   INTERRUPT ***I GUESS***          ?
E7A1 10 06               BPL    SYIRQA
E7A3 AD 01 D3            LDA    PORTB   CLEAR INT STATUS
E7A6 6C 04 02            JMP    (VINTER)
E7A9 08          SYIRQA  PHP
E7AA 68                  PLA            SEE IF SOFTWARE BREAK
E7AB 29 10               AND    #$10    B BIT OF P REGISTER
E7AD F0 03               BEQ    SYIRQB
E7AF 6C 06 02            JMP    (VBREAK)
E7B2 68          SYIRQB  PLA
E7B3 40          SYRTI   RTI    UNIDENTIFIED INTERRUPT, JUST RETURN
```

```
                        ;
                        ; NMI HANDLER
                        ;
                        ; DETERMINE CAUSE AND JUMP THRU VECTOR
                        ;
E7B4 2C 0F D4           PNMI    BIT     NMIST
E7B7 10 03                      BPL     PNMI1   SEE IF DISPLAY LIST
E7B9 6C 00 02                   JMP     (VDSLST)
E7BC 48                 PNMI1   PHA
E7BD AD 0F D4                   LDA     NMIST
E7C0 29 20                      AND     #$20    SEE IF RESET
E7C2 F0 03                      BEQ     *+5
E7C4 4C 74 E4                   JMP     WARMSV  GO THRU WARM START JUMP
E7C7 8A                         TXA             SAVE REGISTERS
E7C8 48                         PHA
E7C9 98                         TYA
E7CA 48                         PHA
E7CB 8D 0F D4                   STA     NMIRES  RESET INTERRUPT STATUS
E7CE 6C 22 02                   JMP     (VVBLKI)        JUMP THRU VECTOR
```

```
                           ;
                           ; SYSTEM VBLANK ROUTINE
                           ;
                           ; INC FRAME COUNTER. PROCESS COUNTDOWN TIMERS. EXIT IF I WAS SET. CLEAR I
                           ; SET DLISTL, DLISTH, DMACTL FROM RAM CELLS. DO SOFTWARE REPEAT
                           ;
      E7D1 E6 14           SYSVBL  INC    RTCLOK+2          INC FRAME COUNTER
      E7D3 D0 08                   BNE    SYSVB1
      E7D5 E6 4D                   INC    ATRACT            INCREMENT ATRACT (CAUSES ATRACT WHEN MINUS)
      E7D7 E6 13                   INC    RTCLOK+1
      E7D9 D0 02                   BNE    SYSVB1
      E7DB E6 12                   INC    RTCLOK
      E7DD A9 FE           SYSVB1  LDA    #$FE              (ATRACT) SET DARK MASK TO NORMAL
      E7DF A2 00                   LDX    #0                SET COLRSH TO NORMAL
      E7E1 A4 4D                   LDY    ATRACT            TEST ATRACT FOR NEGATIVE
      E7E3 10 06                   BPL    VBATRA            WHILE POSITIVE, DONT GO INTO ATRACT
      E7E5 85 4D                   STA    ATRACT            IN ATRACT, SO STAY BY STA $FE
      E7E7 A6 13                   LDX    RTCLOK+1          COLOR SHIFT FOLLOWS RTCLOK+1
      E7E9 A9 F6                   LDA    #$F6              SET DARK MASK TO DARK
      E7EB 85 4E           VBATRA  STA    DRKMSK                                                        stage 1
      E7ED 86 4F                   STX    COLRSH
      E7EF A2 00                   LDX    #0      POINT TO TIMER1
      E7F1 20 FD E8                JSR    DCTIMR  GO DECREMENT TIMER1
      E7F4 D0 03                   BNE    SYSVB2  BRANCH IF STILL COUNTING
      E7F6 20 EF E8                JSR    JTIMR1  GO JUMP TO ROUTINE
      E7F9 A5 42           SYSVB2  LDA    CRITIC
      E7FB D0 08                   BNE    XXIT    GO IF CRITICAL SET
      E7FD BA                      TSX            SEE IF I WAS SET
      E7FE BD 04 01                LDA    $104,X  GET STACKED P
      E801 29 04                   AND    #$04    I BIT
      E803 F0 03                   BEQ    SYSVB3  BRANCH IF OK
      E805 4C 3E E9        XXIT    JMP    XITVBL  I WAS SET, EXIT
      E808 58              SYSVB3  CLI
      E809 AD 0D D4                LDA    PENV    Light pen read.
      E80C 8D 35 02                STA    LPENV
      E80F AD 0C D4                LDA    PENH
      E812 8D 34 02                STA    LPENH
      E815 AD 31 02                LDA    SDLSTH
      E818 8D 03 D4                STA    DLISTH
      E81B AD 30 02                LDA    SDLSTL
      E81E 8D 02 D4                STA    DLISTL
      E821 AD 2F 02                LDA    SDMCTL
      E824 8D 00 D4                STA    DMACTL
      E827 AD 6F 02                LDA    GPRIOR  GLOBAL PRIOR
      E82A 8D 1B D0                STA    PRIOR
      E82D A9 08                   LDA    #$08    TURN OFF KEYBOARD SPEAKER
      E82F 8D 1F D0                STA    CONSOL
      E832 A2 08                   LDX    #8
      E834 BD C0 02        SCOLLP  LDA    PCOLR0,X          LOAD COLOR REGISTERS FROM RAM
      E837 45 4F                   EOR    COLRSH            DO COLOR SHIFT
      E839 25 4E                   AND    DRKMSK            AND DARK ATRACT
      E83B 9D 12 D0                STA    COLPM0,X                                                      stage 2
      E83E CA                      DEX
      E83F 10 F3                   BPL    SCOLLP
      E841 AD F4 02                LDA    CHBAS
      E844 8D 09 D4                STA    CHBASE
      E847 AD F3 02                LDA    CHACT
      E84A 8D 01 D4                STA    CHACTL
      E84D A2 02                   LDX    #2      POINT TO TIMER 2
      E84F 20 FD E8                JSR    DCTIMR
```

```
  E852 D0 03                 BNE    SYSVB4  IF DIDNT GO ZERO
  E854 20 F2 E8              JSR    JTIMR2  DO JUMP TO TIMER2 ROUTINE
  E857 A2 02         SYSVB4  LDX    #2      RESTORE X
  E859 E8            SYSVB8  INX
  E85A E8                    INX
  E85B BD 18 02              LDA    CDTMV1,X
  E85E 1D 19 02              ORA    CDTMV1+1,X
  E861 F0 06                 BEQ    SYSVBA
  E863 20 F5 E8              JSR    DCTIMR  DECREMENT AND SET FLAG IF NONZERO
  E866 9D 26 02              STA    CDTMF3-4,X
  E869 E0 08         SYSVBA  CPX    #8      SEE IF DONE ALL 3
  E86B D0 EC                 BNE    SYSVB8  LOOP
                     ; CHECK DEBOUNCE COUNTER
  E86D AD 0F D2              LDA    SKSTAT
  E870 29 04                 AND    #$04    KEY DOWN BIT
  E872 F0 08                 BEQ    SYVB6A  IF KEY DOWN
                     ; KEY UP SO COUNT IT
  E874 AD F1 02              LDA    KEYDEL  KEY DELAY COUNTER
  E877 F0 03                 BEQ    SYVB6A  IF COUNTED DOWN ALREADY
  E879 CE F1 02              DEC    KEYDEL  COUNT IT                      ? is this proper?
                     ; CHECK SOFTWARE REPEAT TIMER
  E87C AD 2B 02      SYVB6A  LDA    SRTIMR
  E87F F0 17                 BEQ    SYSVB7  DOESNT COUNT
  E881 AD 0F D2              LDA    SKSTAT
  E884 29 04                 AND    #$04    CHECK KEY DOWN BIT
  E886 D0 60                 BNE    SYSVB6  BRANCH IF NO LONGER DOWN
  E888 CE 2B 02              DEC    SRTIMR  COUNT FRAME OF KEY DOWN
  E88B D0 08                 BNE    SYSVB7  BRANCH IF NOT RUN OUT
                     ; TIMER RAN OUT - RESET AND SIMULATE KEYBOARD IRQ
  E88D A9 06                 LDA    #SRTIM2 TIMER VALUE (1/10 SECOND).
  E88F 8D 2B 02              STA    SRTIMR  SET TIMER
  E892 AD 09 D2              LDA    KBCODE  GET THE KEY
  E895 8D FC 02              STA    CH      PUT INTO CH
                     ; READ GAME CONTROLLERS
  E898 A0 01         SYSVB7  LDY    #1
  E89A A2 03                 LDX    #3
  E89C B9 00 D3      STLOOP  LDA    PORTA,Y
  E89F 4A                    LSR    A
  E8A0 4A                    LSR    A
  E8A1 4A                    LSR    A
  E8A2 4A                    LSR    A
  E8A3 9D 78 02              STA    STICK0,X        STORE JOYSTICK
  E8A6 CA                    DEX
  E8A7 B9 00 D3              LDA    PORTA,Y
  E8AA 29 0F                 AND    #$F
  E8AC 9D 78 02              STA    STICK0,X        STORE JOYSTICK
  E8AF CA                    DEX
  E8B0 88                    DEY
  E8B1 10 E9                 BPL    STLOOP

  E8B3 A2 03                 LDX    #3
  E8B5 BD 10 D0      STRL    LDA    TRIG0,X MOVE JOYSTICK TRIGGERS
  E8B8 9D 84 02              STA    STRIG0,X
  E8BB BD 00 D2              LDA    POT0,X  MOVE POT VALUES
  E8BE 9D 70 02              STA    PADDL0,X
  E8C1 BD 04 D2              LDA    POT4,X
  E8C4 9D 74 02              STA    PADDL4,X
  E8C7 CA                    DEX
  E8C8 10 EB                 BPL    STRL
  E8CA 8D 0B D2              STA    POTGO   START POTS FOR NEXT TIME
```

```
                         ;
EBCD A2 06                    LDX     #6
EBCF A0 03                    LDY     #3
EBD1 B9 78 02         PTRLP   LDA     STICK0,Y         TRANSFER BITS FROM JOYSTICKS
EBD4 4A                       LSR     A                TO PADDLE TRIGGERS
EBD5 4A                       LSR     A
EBD6 4A                       LSR     A
EBD7 9D 7D 02                 STA     PTRIG1,X
EBDA A9 00                    LDA     #0
EBDC 2A                       ROL     A
EBDD 9D 7C 02                 STA     PTRIG0,X
EBE0 CA                       DEX
EBE1 CA                       DEX
EBE2 88                       DEY
EBE3 10 EC                    BPL     PTRLP
                         ;
EBE5 6C 24 02                 JMP     (VVBLKD)         GO TO DEFERRED VBLANK ROUTINE
00EB                  SV7H    =       SYSVB7/256
0098                  SV7L    =       -256*SV7H+SYSVB7
EBE8 A9 00            SYSVBB  LDA     #0
EBEA 8D 2B 02                 STA     SRTIMR  ZERO TIMER
EBED F0 A9                    BEQ     SYSVB7  UNCOND
EBEF 6C 26 02         JTIMR1  JMP     (CDTMA1)
EBF2 6C 28 02         JTIMR2  JMP     (CDTMA2)
                         ;
                         ; SUBROUTINE TO DECREMENT A COUNTDOWN TIMER
                         ; ENTRY X=OFFSET FROM TIMER 1
                         ; EXIT A,P=ZERO IF WENT ZERO, FF OTHERWISE
                         ;
EBF5 BC 18 02         DCTIMR  LDY     CDTMV1,X         LO BYTE
EBF8 D0 08                    BNE     DCTIM1  NONZERO, GO DEC IT
EBFA BC 19 02                 LDY     CDTMV1+1,X       SEE IF BOTH ZERO
EBFD F0 10                    BEQ     DCTXF   YES, EXIT NONZERO
EBFF DE 19 02                 DEC     CDTMV1+1,X       DEC HI BYTE
EC02 DE 18 02         DCTIM1  DEC     CDTMV1,X         DEC LO BYTE
EC05 D0 08                    BNE     DCTXF
EC07 BC 19 02                 LDY     CDTMV1+1,X
EC0A D0 03                    BNE     DCTXF
EC0C A9 00                    LDA     #0               WENT ZERO, RETURN ZERO
EC0E 60                       RTS
EC0F A9 FF            DCTXF   LDA     #$FF    RETURN NONZERO
EC11 60                       RTS
```

```
                    ;
                    ; SUBROUTINE TO SET VERTICAL BLANK VECTORS AND TIMERS
                    ; ENTRY X=HI,Y=LO BYTE TO SET
                    ;       A= 1-5 TIMERS 1-5
                    ;          6 IMM VBLANK
                    ;          7 DEF VBLANK
                    ;
E912 0A             SETVBL ASL   A        MUL BY 2
E913 8D 2D 02              STA   INTEMP
E916 A9 00                 LDA   #$00     DLIST OFF BUT VBLANK OFF***FOR NOW***
E918 8D 0E D4              STA   NMIEN
E91B 8A                    TXA
E91C AE 2D 02              LDX   INTEMP
E91F 9D 17 02              STA   CDTMV1-1,X
E922 98                    TYA
E923 9D 16 02              STA   CDTMV1-2,X
E926 A9 40                 LDA   #$40     DLIST OFF***FOR NOW***, VBLANK ON
E928 8D 0E D4              STA   NMIEN
E92B 2C 0F D4              BIT   NMIST    SEE IF A VBLANK HAPPENED
E92E 50 0D                 BVC   SETVBI   IF NOT, GO EXIT
                    ; SIMULATE VERTICAL BLANK NMI
E930 A9 E9                 LDA   #SVIH
E932 48                    PHA            PUSH PC ON STACK AS IF INTERRUPT
E933 A9 3D                 LDA   #SVIL
E935 48                    PHA
E936 08                    PHP            DUMMY P
E937 48                    PHA            DUMMY A
E938 48                    PHA            DUMMY X
E939 48                    PHA            DUMMY Y
E93A 6C 22 02              JMP   (VVBLKI) GO THRU VBLANK VECTOR
E93D 60             SETVBI RTS
00E9                SVIH   =     SETVBI/256
003D                SVIL   =     -256*SVIH+SETVBI
                    ;
                    ; EXIT FROM VERTICAL BLANK
                    ;
E93E 68             XITVBL PLA            UNSTACK Y
E93F A8                    TAY
E940 68                    PLA            UNSTACK X
E941 AA                    TAX
E942 68                    PLA            UNSTACK A
E943 40                    RTI            AND GO BACK FROM WHENCE.
00E6                PIRQH  =     PIRQ/256
00F3                PIRQL  =     -256*PIRQH+PIRQ
00E7                PNMIH  =     PNMI/256
00B4                PNMIL  =     -256*PNMIH+PNMI
                    ;SPARE BYTE OR MODULE TOO LONG FLAG
E944                CRNTP5 =     *
                           *=$14
0014 00             INTSPR .BYTE $E944-CRNTP5  INTHV IS TOO LONG
                           .END
```

```
INTERRUPT HANDLER

            SYMBOL TABLE
ABUFPT  001C    ADRESS  0064    AFP     D800    ALLPOT  D208
ANTIC   D400    APPEND  0001    APPMHI  000E    ATACHR  02FB
ATAN    BE43    ATRACT  004D    AUDC1   D201    AUDC2   D203
AUDC3   D205    AUDC4   D207    AUDCTL  D208    AUDF1   D200
AUDF2   D202    AUDF3   D204    AUDF4   D206    [illegible]  00BA
BADMOD  0091    BFENHI  0035    BFENLO  0034    BITMSK  006E
BLIM    028A    BLKBDV  E471    BOOT?   0009    BOOTAD  0242
BOTSCR  02BF    BPTR    003D    BRKABT  0080    BRKKEY  0011
BUFADR  0015    BUFCNT  006B    BUFRFL  0038    BUFRHI  0033
BUFRLO  0032    BUFSTR  006C    CASBUF  03FD    CASETV  E440
CASFLG  030F    CASINI  0002    CASORG  EFA1    CASSBT  004B
CASSET  0003    CAUX1   023C    CAUX2   023D    CBAUDH  02EF
CBAUDL  02EE    CCOMND  023B    CDEVIC  023A    CDTMA1  0226
CDTMA2  0228    CDTMF3  022A    CDTMF4  022C    CDTMF5  022E
CDTMV1  0218    CDTMV2  021A    CDTMV3  021C    CDTMV4  021E
CDTMV5  0220    CH      02FC    CH1     02F2    CHACT   02F3
CHACTL  D401    CHAR    02FA    CHBAS   02F4    CHBASE  D409
CHKERR  003B    CHKSNT  003C    CHKSUM  0031    CHRORG  E000
CIOCHR  002F    CIOINV  E6E4    CIOORG  E4A6    CIOV    E456
CIX     00F2    CKEY    004A    CLOSE   000C    COLAC   0072
COLBK   D01A    COLCRS  0055    COLDST  0244    COLDSV  E477
COLINC  0074    COLOR0  02C4    COLOR1  02C5    COLOR2  02C6
COLOR3  02C7    COLOR4  02C8    COLPF0  D016    COLPF1  D017
COLPF2  D018    COLPF3  D019    COLPM0  D012    COLPM1  D013
COLPM2  D014    COLPM3  D015    COLRSH  004F    CONSOL  D01F
COS     BD73    COUNTR  007E    CR      009B    CRETRY  0036
CRITIC  0042    CRNTPC  E944    CRSINH  02F0    CSRROR  008D
CSOPIV  E47D    CSTAT   0288    CTIA    D000    DAUX1   030A
DAUX2   030B    DBSECT  0241    DBUFHI  0305    DBUFLO  0304
DBYTHI  0309    DBYTLO  0308    DCB     0300    DCOMND  0302
DCSINI  E762    DCTIMR  EDE5    DELAE   E90F    DDEVIC  0300
DEGFLG  00?D    DEGCNT  0006    DELETE  0021    DELTAC  0077
DELTAR  0076    DERROR  0090    DFLAGS  0240    DIGRT   00F1
DINDEX  0057    DIRECT  0002    DISK    0044    DISKIV  E450
DISPLY  0059    DLISTH  D403    DLISTL  D402    DMACTL  D400
DMASK   02A0    DRACK   0088    DOSINI  000C    DOSVEC  000A
DRAWLN  00LL    DRETRY  0037    DRKMSK  004E    DSKFMS  0018
DSKINV  E453    DSKORG  EDEA    DSKTIM  0246    DSKUTL  001A
DSPFLG  02FE    DSTAT   004C    DSTATS  0303    DTIMLO  0306
DUNIT   0301    DUNUSE  0307    DVSTAT  02EA    EDITRV  E400
ENDP    0091    ENDPT   0074    EOFFRM  0088    ERRFLG  023F
ERCFLG  0292    ESIGN   00EF    EXP     DDC0    EXP10   DDCC
FADD    DA66    FASC    D8E6    FCHRFL  00F0    FDIV    DB28
FEOF    003F    FILDAT  02FD    FILFLG  02B7    FILLDR  0013
FLD0P   DD8D    FLD0R   DD89    FLD1P   DD9C    FLD1R   DD98
FLPTR   00FC    FMOVE   DDB6    FMSZPG  0043    FMUL    DADB
[illegible]  0092    FORMAT  02DC    FPI     D?02    [illegible]  0006
[illegible]  0556    [illegible]  [illegible]    FPTR2   00FE    FR0     00D4
FR1     00E0    FR2     00E6    FRE     00DA    FREQ    0040
FRMERR  009C    FRX     00EC    FSCR    05E6    FSCR1   05EC
FSTOP   DDAB    FSTOR   DDA7    FSUB    DA60    FTYPE   DDDE
GETCHR  F6??    GETREC  [illegible]    GLBABS  02E0    GPRIOR  026F
GRACTL  D01D    GRAFM   D011    GRAFP0  D00D    GRAFP1  D00E
GRAFP2  D00F    GRAFP3  D010    HATABS  031A    HITCLR  D01E
HOLD1   0051    HOLD2   029F    HOLD3   029D    HOLD4   02BC
HOLD5   02A0    HOLDCH  007C    HPOSM0  D004    HPOSM1  D005
HPOSM2  D006    HPOSM3  D007    HPOSP0  D000    HPOSP1  D001
HPOSP2  D002    HPOSP3  D003    HSCROL  D404    ICAX1   034A
ICAX1Z  002A    ICAX2   034B    ICAX2Z  002B    ICAX3   034C
```

```
HPOSP2  D002    HPOSP3  D003    HSCROL  D404    ICAX1   034A
ICAX1Z  002A    ICAX2   034B    ICAX2Z  002B    ICBAH   0345
```

INTERRUPT HANDLER

```
ICBAHZ  0025    ICBAL   0344    ICBALZ  0024    ICBLH   0349
ICBLHZ  0029    ICBLL   0348    ICBLLZ  0028    ICCOM   0342
ICCOMT  0017    ICCOMZ  0022    ICDNO   0341    ICDNOZ  0021
ICHID   0340    ICHIDZ  0020    ICIDNO  002E    ICPTH   0347
ICPTHZ  0027    ICPTL   0346    ICPTLZ  0026    ICSPR   034C
ICSPRZ  002C    ICSTA   0343    ICSTAZ  0023    IFP     D9AA
IHINIT  EAD8    INBUFF  00F3    INSCLN  0020    INRDAT  007D
INTABS  0200    INTEMP  022D    INTINV  E46B    INTORG  E4D5
INTSPR  0014    INTVBS  0040    INVFLG  02B6    IOCB    0340
IOCBAS  0020    IOCBSZ  0010    IOCFRE  00FF    IRQEN   D20E
IRQST   D20E    JTIMR1  EBFF    JTIMR2  EBF2    KBCODE  D209
KBD     0040    KBDORG  FEF6    KEYBDV  E420    KEYDEL  02F1
LBFEND  05FF    LBPR1   057E    LBPR2   057F    LBUFF   0580
LEDGE   0002    LINBUF  0247    LINZBS  0000    LMARGN  0052
LOADFL  0023    LOG     DECD    LOG10   DED1    LOGCOL  0063
LOGMAP  02B2    LPENH   0234    LPENV   0235    M0PF    D000
M0PL    D008    M1PF    D001    M1PL    D009    M2PF    D002
M2PL    D00A    M3PF    D003    M3PL    D00B    MAXDEV  0021
MAXIOC  0080    MEMLO   02E7    MEMTOP  02E5    MLTTMP  0066
MODEM   0040    MONORG  E0E3    MXDADD  0010    NEWCOL  0061
NEWROW  0060    NMIEN   D40E    NMIRES  D40F    NMIST   D40F
NOCKSM  003C    NONDEV  0082    NOTE    0026    NOTOPN  0085
NSIGN   00EE    NVALID  0086    OLDADR  005E    OLDCHR  005D
OLDCOL  005B    OLDROW  005A    OPEN    0003    OPNIN   0004
OPNINO  000C    OPNOT   0008    OPNTMP  0065    OVRRUN  008E
P0PF    D004    P0PL    D00C    P1PF    D005    P1PL    D00D
P2PF    D006    P2PL    D00E    P3PF    D007    P3PL    D00F
PACTL   D302    PADDL0  0270    PADDL1  0271    PADDL2  0272
PADDL3  0273    PADDL4  0274    PADDL5  0275    PADDL6  0276
PADDL7  0277    PBCTL   D303    PBPNT   001D    PBUFSZ  001E
PCOLR0  02C0    PCOLR1  02C1    PCOLR2  02C2    PCOLR3  02C3
PENH    D40C    PENV    D40D    PIA     D300    PIRQ    E6F3
PIRQH   00E6    PTRBL   00F3    PLYARG  05E0    PLYEVL  DD40
PMBASE  D407    PNMI    E7B4    PNMI1   E7BC    PNMOTH  00E2
PNMOD   0014    POINT   0025    POKEY   D200    POKMSK  0010
PORTA   D300    PORTB   D301    POT0    D200    POT1    D201
POT2    D202    POT3    D203    POT4    D204    POT5    D205
POT6    D206    POT7    D207    POTGO   D20B    PRINTD  0050
PRINTV  E430    PRIOR   D01B    PRNBUF  03C0    PRNORM  EE78
PRVOPN  0051    PTEMP   001F    PTIMOT  001C    PTRIG0  027C
PTRIG1  027D    PTRIG2  027E    PTRIG3  027F    PTRIG4  0280
PTRIG5  0281    PTRIG6  0282    PTRIG7  0283    PTRLP   E8D1
PUTCHR  000B    PUTREC  0009    RADFLG  00FB    RADON   0000
RAMLO   0004    RAMSIZ  02E4    RAMTOP  006A    RANDOM  D20A
RBLOKV  E47A    RDONLY  0087    RECVDN  0039    REDONE  0027
RFVAME  0020    RMARGN  0053    ROWAC   0070    ROWCRS  0054
ROWINC  0009    RTCLOK  0012    SAVADR  0068    SAVIO   0316
SAVMSC  0058    SCOLAP  F1C8    SCPFLT  0045    SCRENV  E410
SCRFLG  02BB    SCRMEM  D089    SDLSTH  0231    SDLSTL  0230
SDMCTL  022F    SENDEV  E460    SERIN   D20D    SEROUT  D20D
SETVBI  E7D0    SETVBL  E912    SETVBV  E45C    SHFAMT  006F
SHFLOK  02BE    SIN     BD81    SIOINV  E465    SIOSUB  E948
SIOV    E459    SIZEM   D00C    SIZEP0  D008    SIZEP1  D009
SIZEP2  D00A    SIZEP3  D00B    SKCTL   D20F    SKRES   D20A
SKSTAT  D20F    SOUNDR  0041    SPFLAG  000E    SQR     BF81
SQUAM2  0085    SRTIMR  022B    SSFLAG  02FF    SSKCTL  0232
STACKP  0318    STAFIB  00BB    STATUS  0030    STICK0  0278
STICK1  0279    STICK2  027A    STICK3  027B    STIMER  D209
STLOOP  EB5C    STRIG0  0284    STRIG1  0285    STRIG2  0286
STRIG3  0287    STRL    1000    SUBTMP  029E    SXPCFF  0001
```

```
INTERRUPT HANDLER

SV1H    00E9    SV1L    0030    SV7H    00E8    SV7L    0098
SWPFLG  007B    SYIRQ   E6F6    SYIRQ1  E708    SYIRQ2  E71F
SYIRQ3  E737    SYIRQ4  E74A    SYIRQ5  E75A    SYIRQ6  E767
SYIRQ7  E779    SYIRQ8  E793    SYIRQ9  E79E    SYIRQA  E7A9
SYIRQB  E7B2    SYRTI   E7B3    SYSVB1  E7DD    SYSVB2  E7E9
SYSVB3  E800    SYSVB4  E857    SYSVB6  E8EB    SYSVB7  E898
SYSVBA  E869    SYSVBB  E859    SYSVBL  E7D1    SYSVBV  E45F
SYVB6A  E87C    TABMAP  02A3    TEMP    023E    TEMP1   0312
TEMP2   0314    TEMP3   0315    TIMER1  030C    TIMER2  0310
TIMFLG  0317    TIMOUT  008A    TINDEX  0293    TMPCHR  0050
TMPCOL  02B9    TMPLBT  02A1    TMPROW  02B8    TMPX1   029C
TRANS2  0006    TRIG0   D010    TRIG1   D011    TRIG2   D012
TRIG3   D013    TRNRCD  0087    TSTAT   0319    TSTDAT  0007
TXTCOL  0291    TXTMSC  0294    TXTOLD  0296    TXTROW  0290
UNLOCK  0024    USAREA  0480    VBATRA  E7EB    VBREAK  0206
VCOUNT  D40B    VCTABL  E480    VDELAY  D01C    VDSLST  0200
VECTBL  E400    VIMIRQ  0216    VINTER  0204    VKEYBD  0208
VPRCED  0202    VSCROL  D405    VSERIN  020A    VSEROC  020E
VSEROR  020C    VTIMR1  0210    VTIMR2  0212    VTIMR4  0214
VVBLKD  0224    VVBLKI  0222    WARMST  0008    WARMSV  E474
WMODE   0289    WRONLY  0083    WSYNC   D40A    XITVBL  E93E
XITVBV  E462    XMTDON  003A    XXIT    EB05    ZIOCB   0020
ZTEMP1  00F5    ZTEMP3  00F9    ZTEMP4  00F7
```

```
SIO  ( SERIAL BUS INPUT/OUTPUT CONTROLLER )
```

```
                 TITLE  'SIO  ( SERIAL BUS INPUT/OUTPUT CONTROLLER )'
;               COLLEEN OPERATING SYSTEM
;
;               SIO  ( SERIAL BUS INPUT/OUTPUT CONTROLLER )
;               WITH SOFTWARE BAUD RATE CORRECTION ON CASSETTE
;
;
;                    AL MILLER       3-APR-79
;
;
;THIS MODULE HAS ONE ENTRY POINT.  IT IS CALLED BY THE DEVICE
;HANDLERS.  IT INTERPRETS A PREVIOUSLY ESTABLISHED DEVICE CONTROL
;BLOCK (STORED IN GLOBAL RAM) TO ISSUE COMMANDS
;TO THE SERIAL BUS TO CONTROL TRANSMITTING AND RECEIVING DATA
;
;
;
;
```

```
                        ;EQUATES
                        ;
                        ;DCB DEVICE BUS ID NUMBERS
      0030              FLOPPY  =  $30
      0040              PRINTR  =  $40
      0060              CASSET  =  $60
                        ;
                        ;
                        ;BUS COMMANDS
                        ;
      0052              READ    =  'R
      0057              WRITE   =  'W
      0053              STATIS  =  'S
      0021              FORMAT  =  '!
                        ;
                        ;
                        ;COMMAND AUX BYTES
                        ;
      0053              SIDWAY  =  'S   PRINT 16 CHARACTERS SIDEWAYS
      004E              NORMAL  =  'N   PRINT 40 CHARACTERS NORMALLY
      0044              DOUBLE  =  'D   PRINT 20 CHARACTERS DOUBLE WIDE
      0050              PLOT    =  'P   PLOT MODE
                        ;
                        ;
                        ;BUS RESPONSES
                        ;
      0041              ACK     =  'A   DEVICE ACKNOWLEDGES INFORMATION
      004E              NACK    =  'N   DEVICE DID NOT UNDERSTAND
      0043              COMPLT  =  'C   DEVICE SUCCESSFULLY COMPLETED OPERATION
      0045              ERROR   =  'E   DEVICE INCURRED AN ERROR IN AN ATTEMPTED OPERATION
                        ;
                        ;
                        ;MISCELLANEOUS EQUATES
                        ;
      0028              B192LO  =  $28  19200 BAUD RATE POKEY COUNTER VALUES (LO BYTE)
      0000              B192HI  =  $00                                        (HI BYTE)
      00CC              B600LO  =  $CC    600 BAUD                            (LO BYTE)
      0005              B600HI  =  $05                                        (HI BYTE)
      0005              HITONE  =  $05  FSK HI FREQ POKEY COUNTER VALUE (5326 HZ)
      0007              LOTONE  =  $07  FSK LO FREQ POKEY COUNTER VALUE (3995 HZ)
                        ;
      0064              WIRGLO  =  100  WRITE INTER RECORD GAP (IN 1/60 SEC)
      0000              WIRGHI  =  0
      0078              RIRGLO  =  120  READ INTER RECORD GAP (IN 1/60 SEC)
      0000              RIRGHI  =  0
      000F              WSIRG   =  15   SHORT WRITE INTER RECORD GAP
      000A              RSIRG   =  10   SHORT READ INTER RECORD GAP
                        ;
      0034              NCOMLO  =  $34  PIA COMMAND TO LOWER NOT COMMAND LINE
      003C              NCOMHI  =  $3C  PIA COMMAND TO RAISE NOT COMMAND LINE
      0034              MOTRGO  =  $34  PIA COMMAND TO TURN ON CASSETTE MOTOR
      003C              MOTRST  =  $3C  PIA COMMAND TO TURN OFF MOTOR
                        ;
      0002              TEMPHI  =  TEMP/256     ADDRESS OF TEMP CELL (HI BYTE)
      003E              TEMPLO  =  -256*TEMPHI+TEMP                 (LO BYTE)
      0002              CBUFHI  =  CDEVIC/256   ADDRESS OF COMMAND BUFFER (HI BYTE)
      003A              CBUFLO  =  -256*CBUFHI+CDEVIC                      (LO BYTE)
                        ;
      000D              CRETRI  =  13   NUMBER OF COMMAND FRAME RETRIES
      0001              DRETRI  =  1    NUMBER OF DEVICE RETRIES
```

SIO (SERIAL BUS INPUT/OUTPUT CONTROLLER)

```
     0002                CTIMLO = 2     COMMAND FRAME ACK TIME OUT (LO BYTE)
     0000                CTIMHI = 0     COMMAND FRAME ACK TIME OUT (HI BYTE)
                         ;
                         ;
     00EB                JTADRH = JTIMER/256      HI BYTE OF JUMP TIMER ROUTINE ADDR
     00F0                JTADRL = -256*JTADRH+JTIMER
                         ;
```

SIO (SERIAL BUS INPUT/OUTPUT CONTROLLER)

```
                        ;       SIO
                        ;
                        ;
                                *=SIOV
E459 4C 59 E9                   JMP     SIO     SIO ENTRY POINT
                        ;
                                *=SIOINV
E465 4C 44 E9                   JMP     SIOINT  SIO INITIALIZATION ENTRY POINT
                        ;
                                *=SENDEV
E468 4C EA ED                   JMP     SENDEN  SEND ENABLE ENTRY POINT
                        ;
                                *=VCTABL-INTABS+VSERIN
                        ;
E48A 11 EB ✓                    .WORD   ISRSIR  VSERIN
E48C 90 EA ✓                    .WORD   ISRODN  VSEROR
E48E D1 EA ✓                    .WORD   ISRTD   VSEROC
                        ;
                        ;
                        ;
                                *=SIOORG
                        ;
                        ;SIO INITIALIZATION SUBROUTINE
                        ;
E944 A9 3C              SIOINT  LDA     #MOTRST
E946 8D 02 D3                   STA     PACTL   TURN OFF MOTOR
                        ;
E949 A9 3C                      LDA     #NCOMHI
E94B 8D 03 D3                   STA     PBCTL   RAISE NOT COMMAND LINE
                        ;
                        ;
E94E A9 03                      LDA     #3
E950 8D 32 02                   STA     SSKCTL  GET POKEY OUT OF INITIALIZE MODE
E953 85 41                      STA     SOUNDR  INIT POKE ADDRESS FOR ~~QUIET~~ I/O
                                                                           NOISY
E955 8D 0F D2                   STA     SKCTL
                        ;
                        ;
E958 60                         RTS             RETURN
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
E959 BA                 SIO     TSX
E95A 8E 18 03                   STX     STACKP  SAVE STACK POINTER
E95D A9 01                      LDA     #1
E95F 85 42                      STA     CRITIC
                        ;
E961 AD 00 03                   LDA     DDEVIC
E964 C9 60                      CMP     #CASET
E966 D0 03                      BNE     NOTCST  BRANCH IF NOT CASSETTE
E968 4C B4 ED                   JMP     CASENT  OTHERWISE JUMP TO CASSETTE ENTER
                        ;
                        ;ALL DEVICES EXCEPT CASSETTE ARE INTELLIGENT
                        ;
E96B A9 00              NOTCST  LDA     #0
E96D 8D 0F 03                   STA     CASFLG  INIT CASSETTE FLAG TO NO CASSETTE
                        ;
E970 A9 01                      LDA     #DRETRI SET NUMBER OF DEVICE RETRIES
```

```
E952 85 37                  STA     DRETRY
E954 A9 0D         COMMND   LDA     #CRETRI  SET NUMBER OF COMMAND FRAME RETRIES
E956 85 36                  STA     CRETRY
                   ;
                   ;SEND A COMMAND FRAME
                   ;
E958 A9 28         COMFRM   LDA     #B192LO SET BAUD RATE TO 19200
E95A 8D 04 D2               STA     AUDF3
E95D A9 00                  LDA     #B192HI
E95F 8D 06 D2               STA     AUDF4
                   ;
E962 18                     CLC             SET UP COMMAND BUFFER
E963 AD 00 03               LDA     DDEVIC
E966 6D 01 03               ADC     DUNIT
E969 69 FF                  ADC     #$FF    SUBTRACT 1
E96B 8D 3A 02               STA     CDEVIC  SET BUS ID NUMBER
                   ;
E96E AD 02 03               LDA     DCOMND
E971 8D 3B 02               STA     CCOMND  SET BUS COMMAND
                   ;
E974 AD 0A 03               LDA     DAUX1   STORE COMMAND FRAME AUX BYTES 1 AND 2
E977 8D 3C 02               STA     CAUX1
E97A AD 0B 03               LDA     DAUX2
E97D 8D 3D 02               STA     CAUX2   DONE SETTING UP COMMAND BUFFER
                   ;
E980 18                     CLC             SET BUFFER POINTER TO COMMAND FRAME BUFFER
E981 A9 3A                  LDA     #CBUFLO
E983 85 32                  STA     BUFRLO  AND BUFFER END ADDRESS
E985 69 04                  ADC     #4
E987 85 34                  STA     BFENLO
E989 A9 02                  LDA     #CBUFHI
E98B 85 33                  STA     BUFRHI
E98D 85 35                  STA     BFENHI  DONE SETTING UP BUFFER POINTER
                   ;
E98F A9 34                  LDA     #NCOMLO LOWER NOT COMMAND LINE
E991 8D 03 D3               STA     PBCTL
                   ;
E994 20 8E EC               JSR     SENDIN  SEND THE COMMAND FRAME TO A SMART DEVICE
                   ;
E997 AD 3F 02               LDA     ERRFLG
E99A D0 03                  BNE     BADCOM  BRANCH IF AN ERROR RECEIVED
                   ;
E99C 98                     TYA
E99D D0 07                  BNE     ACKREC  BRANCH IF ACK RECEIVED
                   ;
                   ;
E99F C6 36         BADCOM   DEC     CRETRY  A NACK OR TIME OUT OCCURED
E9A1 10 B5                  BPL     COMFRM  SO BRANCH IF ANY RETRIES LEFT
                   ;
E9A3 4C 06 EA               JMP     DERR1   OTHERWISE, JUMP TO RETURN SECTION
                   ;
                   ;
E9A6 AD 03 03      ACKREC   LDA     DSTATS  ACK WAS RECEIVED
E9A9 10 0C                  BPL     WATCOM  BRANCH TO WAIT FOR COMPLETE,
                   ;                         IF THERE IS NO DATA TO BE SENT
                   ;
                   ;
                   ;SEND A DATA FRAME TO PERIPHERAL
                   ;
```

```
E9CB A9 0D               LDA     #CRETRI SET NUMBER OF RETRIES
E9CD 85 36               STA     CRETRY
                  ;
E9CF 20 6E EB            JSR     LDPNTR  LOAD BUFFER POINTER WITH DCB INFORMATION
                  ;
E9D2 20 8E EC            JSR     SENDIN  GO SEND THE DATA FRAME TO A SMART DEVICE
                  ;
E9D5 F0 E8               BEQ     BADCOM  BRANCH IF BAD (timeout/NAK/ACK)
                  ;                      DSERR?
                  ;
                  ;
                  ;WAIT FOR COMPLETE SIGNAL FROM PERIPHERAL
                  ;
E9D7 20 79 EC     WAITCM JSR     STTMOT  SET DDEVICE TIME OUT VALUES IN Y,X
                  ;
E9DA A9 00               LDA     #$00
E9DC 8D 3F 02            STA     ERRFLG  CLEAR ERROR FLAG
                  ;
E9DF 20 9F EC            JSR     WAITER  SET UP TIMER AND WAIT
E9E2 F0 12               BEQ     DERR    BRANCH IF TIME OUT
                  ;
                  ;
                  ;DEVICE DID NOT TIME OUT
                  ;
E9E4 2C 03 03            BIT     DSTATS
E9E7 70 07               BVS     MODATA  BRANCH IF MORE DATA FOLLOWS
                  ;
E9E9 AD 3F 02            LDA     ERRFLG
E9EC D0 18               BNE     DERR1   BRANCH IF AN ERROR OCCURRED
E9EE F0 1D               BEQ     RETURN  OTHERWISE RETURN
                  ;
                  ;
                  ;
                  ;
                  ;RECEIVE A DATA FRAME FROM PERIPHERAL
                  ;
E9F0 20 6E EB     MODATA JSR     LDPNTR  LOAD BUFFER POINTER WITH DCB INFORMATION
                  ;
E9F3 20 E2 EA            JSR     RECEIV  GO RECEIVE A DATA FRAME
                  ;
E9F6 AD 3F 02     DERR   LDA     ERRFLG
E9F9 F0 05               BEQ     NOTERR  BRANCH IF NO ERROR PRECEEDED DATA
                  ;
E9FB AD 19 03            LDA     TSTAT   GET TEMP STATUS
E9FE 85 30               STA     STATUS  STORE IN REAL STATUS
                  ;
                  ;
EA00 A5 30        NOTERR LDA     STATUS
EA02 C9 01               CMP     #SUCCES
EA04 F0 07               BEQ     RETURN  BRANCH IF COMPLETELY SUCCESSFUL
                  ;
EA06 C6 37        DERR1  DEC     DRETRY
EA08 30 03               BMI     RETURN  BRANCH IF OUT OF DEVICE RETRIES
                  ;
EA0A 4C 74 E9            JMP     COMMND  OTHERWISE, ONE   MORE   TIME
                  ;
                  ;
                  ;
                  ;
EA0D 20 63 EC     RETURN JSR     SENDDS  DISABLE POKEY INTERRUPTS
```

```
   EA10 A9 00                LDA    #0
   EA12 85 42                STA    CRITIC
   EA14 A4 30                LDY    STATUS   RETURN STATUS IN Y
   EA16 8C 03 03             STY    DSTATS   AND THE DCB STATUS WORD
   EA19 60                   RTS    RETURN
                    ;
                    ;
                    ;
                    ;
                    ;WAIT SUBROUTINE
                    ;
                    ;WAITS FOR COMPLETE OR ACK
                    ;RETURNS Y=$FF IF SUCCESSFUL, Y=$00 IF NOT
                    ;
   EA1A A9 00       WAIT     LDA    #$00
   EA1C 8D 3F 02             STA    ERRFLG   CLEAR ERROR FLAG
                    ;
   EA1F 18                   CLC             LOAD BUFFER POINTER WITH ADDRESS
   EA20 A9 3E                LDA    #TEMPLO  OF TEMPORARY RAM CELL
   EA22 85 32                STA    BUFRLO
   EA24 69 01                ADC    #1
   EA26 85 34                STA    BFENLO   ALSO SET BUFFER END +1 ADDRESS
   EA28 A9 02                LDA    #TEMPHI
   EA2A 85 33                STA    BUFRHI
   EA2C 85 35                STA    BFENHI   DONE LOADING POINTER
                    ;
   EA2E A9 FF                LDA    #$FF
   EA30 85 3C                STA    NOCKSM   SET NO CHECKSUM FOLLOWS DATA FLAG
                    ;
   EA32 20 E2 EA             JSR    RECEIV   GO RECEIVE A BYTE
                    ;
   EA35 A0 FF                LDY    #$FF     ASSUME SUCCESS
   EA37 A5 30                LDA    STATUS
   EA39 C9 01                CMP    #SUCCES
   EA3B D0 19                BNE    NWOK     BRANCH IF IT DID NOT WORK OK
                    ;
                    ;
                    ;
                    ;
   EA3D AD 3E 02    WOK      LDA    TEMP     MAKE SURE THE BYTE SUCCESSFULLY RECEIVED
   EA40 C9 41                CMP    #ACK     WAS ACTUALLY AN ACK OR COMPLETE
   EA42 F0 21                BEQ    GOOD
   EA44 C9 43                CMP    #COMPLT
   EA46 F0 1D                BEQ    GOOD
                    ;
   EA48 C9 45                CMP    #ERROR
   EA4A D0 06                BNE    NOTDER   BRANCH IF DEVICE DID NOT SEND BACK
                    ;                                    A DEVICE ERROR CODE
   EA4C A9 90                LDA    #DERROR
   EA4E 85 30                STA    STATUS           SET DEVICE ERROR STATUS
   EA50 D0 04                BNE    NWOK
                    ;
   EA52 A9 8B       NOTDER   LDA    #DNACK   OTHERWISE SET NACK STATUS
   EA54 85 30                STA    STATUS
                    ;
   EA56 A5 30       NWOK     LDA    STATUS
   EA58 C9 8A                CMP    #TIMOUT
   EA5A F0 07                BEQ    BAD      BRANCH IF TIME OUT
                    ;
   EA5C A9 FF                LDA    #$FF
```

```
EA5E 8D 3F 02               STA     ERRFLG  SET SOME ERROR FLAG
EA61 D0 02                  BNE     GOOD    RETURN WITH OUT SETTING Y = 0
                   ;
EA63 A0 00         BAD      LDY     #0
                   ;
EA65 A5 30         GOOD     LDA     STATUS
EA67 8D 19 03               STA     TSTAT
EA6A 60                     RTS             RETURN
                   ;
                   ;
                   ;
                   ;
                   ;
                   ;SEND SUBROUTINE
                   ;
                   ;SENDS A BUFFER OF BYTES OUT OVER THE SERIAL BUS
                   ;
                   ;
EA6B A9 01         SEND     LDA     #SUCCES ASSUME SUCCESS
EA6D 85 30                  STA     STATUS
                   ;
EA6F 20 F6 EC               JSR     SENDEN  ENABLE SENDING
                   ;
EA72 A0 00                  LDY     #0
EA74 84 31                  STY     CHKSUM  CLEAR    CHECK SUM
EA76 84 3B                  STY     CHKSNT           CHECKSUM SENT FLAG
EA78 84 3A                  STY     XMTDON           TRANSMISSION DONE FLAG
                   ;
                   ;
EA7A B1 32                  LDA     (BUFRLO),Y       PUT FIRST BYTE FROM BUFFER
EA7C 8D 0D D2               STA     SEROUT           INTO THE SERIAL OUTPUT REGISTER
                   ;
                   ;
EA7F 85 31                  STA     CHKSUM  PUT IT IN CHECKSUM
                   ;
EA81 A5 11         NOTDON   LDA     BRKKEY
EA83 D0 03                  BNE     NTBRK0
EA85 4C A4 ED               JMP     BROKE   JUMP IF BREAK KEY PRESSED
                   ;
EA88 A5 3A         NTBRK0   LDA     XMTDON  LOOP UNTIL TRANSMISSION IS DONE
EA8A F0 F5                  BEQ     NOTDON
                   ;
EA8C 20 A3 EC               JSR     SENDDS  DISABLE SENDING
                   ;
EA8F 60                     RTS             RETURN
                   ;
                   ;
                   ;
                   ;
                   ;
                   ;
                   ;OUTPUT DATA NEEDED INTERRUPT SERVICE ROUTINE
                   ;
EA90 98            ISRODN   TYA
EA91 48                     PHA             SAVE Y REG ON STACK
                   ;
EA92 E6 32                  INC     BUFRLO  INCREMENT BUFFER POINTER
EA94 D0 02                  BNE     NOWRPO
EA96 E6 33                  INC     BUFRHI
```

SIO (SERIAL INPUT/OUTPUT CONTROLLER)

```
 EA98 A5 33          ISRODN LDA    BUFRHI  CHECK IF PAST END OF BUFFER
 EA9A C5 35                 CMP    BFENHI
 EA9C 90 22                 BCC    NOTEND
 EA9E A5 32                 LDA    BUFRLO
 EAA0 C5 34                 CMP    BFENLO
 EAA2 90 1C                 BCC    NOTEND  BRANCH IF NOT PAST END OF BUFFER
                     ;
 EAA4 A5 3B                 LDA    CHKSNT
 EAA6 D0 0B                 BNE    RELONE  BRANCH IF CHECKSUM ALREADY SENT
                     ;
 EAA8 A5 31                 LDA    CHKSUM
 EAAA 8D 0D D2              STA    SEROUT  SEND CHECK SUM
 EAAD A9 FF                 LDA    #$FF
 EAAF 85 3B                 STA    CHKSNT  SET CHECKSUM SENT FLAG
 EAB1 D0 09                 BNE    CHKDON
                     ;
 EAB3 A5 10          RELONE LDA    POKMSK  ENABLE TRANSMIT DONE INTERRUPT
 EAB5 09 08                 ORA    #$08
 EAB7 85 10                 STA    POKMSK  ?
 EAB9 8D 0E D2              STA    IRQEN
                     ;
 EABC 68             CHKDON PLA
 EABD A8                    TAY            RESTORE Y REG
 EABE 68                    PLA            RETURN FROM INTERRUPT
 EABF 40                    RTI
                     ;
                     ;
 EAC0 A0 00          NOTEND LDY    #0
 EAC2 B1 32                 LDA    (BUFRLO),Y      PUT NEXT BYTE FROM BUFFER
 EAC4 8D 0D D2              STA    SEROUT          INTO THE SERIAL OUTPUT REGISTER
                     ;
 EAC7 18                    CLC            ADD IT TO CHECKSUM
 EAC8 65 31                 ADC    CHKSUM
 EACA 69 00                 ADC    #0
 EACC 85 31                 STA    CHKSUM
                     ;
 EACE 4C BC EA              JMP    CHKDON  GO RETURN
                     ;
                     ;
                     ;
                     ;
                     ;
                     ; TRANSMIT DONE INTERRUPT SERVICE ROUTINE
                     ;
 EAD1 A5 3B          ISRXD  LDA    CHKSNT
 EAD3 F0 0B                 BEQ    ENDEY   BRANCH IF CHECKSUM NOT YET SENT
                     ;
 EAD5 85 3A                 STA    XMTDON  OTHERWISE SET TRANSMISSION DONE FLAG
                     ;
 EAD7 A5 10                 LDA    POKMSK  DISABLE TRANSMIT DONE INTERRUPT
 EAD9 29 F7                 AND    #$F7
 EADB 85 10                 STA    POKMSK  ?
 EADD 8D 0E D2              STA    IRQEN
                     ;
 EAE0 68             ENDEY  PLA            RETURN FROM INTERRUPT
 EAE1 40                    RTI
                     ;
                     ;
                     ;
```

SIO (SERIAL BUS INPUT/OUTPUT CONTROLLER)

```
                          ;
                          ;
                          ;
                          ;
                          ;
                          ;RECEIVE SUBROUTINE
                          ;
        EAE2 A9 00        RECEIV  LDA     #0
                          ;
        EAE4 AC 0F 03             LDY     CASFLG
        EAE7 D0 02                BNE     NOCLR   BRANCH IF CASSETTE
                          ;
        EAE9 85 31                STA     CHKSUM  CLEAR   CHKSUM
        EAEB 85 38        NOCLR   STA     BUFRFL          BUFFER FULL FLAG
        EAED 85 39                STA     RECVDN          RECEIVE DONE FLAG
                          ;
                          ;
                          ;
        EAEF A9 01                LDA     #SUCCES
        EAF1 85 30                STA     STATUS          SET GOOD STATUS FOR DEFAULT CASE.
        EAF3 20 1F EC             JSR     RECVEN          DO RECEIVE ENABLE
        EAF6 A9 3C                LDA     #NCOMHI         COMMAND FRAME HI COMMAND
        EAF8 8D 03 D3             STA     PBCTL           STORE IN PIA
        EAFB A5 11        CHRTIM  LDA     BRKKEY
        EAFD D0 03                BNE     NTBRK1
        EAFF 4C A4 ED             JMP     BROKE   JUMP IF BREAK KEY PRESSED
                          ;
        EB02 AD 17 03     NTBRK1  LDA     TIMFLG          NO,
        EB05 F0 05                BEQ     TOUT            IF TIMEOUT, GO SET ERROR STATUS
        EB07 A5 39                LDA     RECVDN
        EB09 F0 F0                BEQ     CHRTIM          DONE ?
        EB0B 60           GOBACK  RTS
        EB0C A9 8A        TOUT    LDA     #TIMOUT         YES,
        EB0E 85 30                STA     STATUS          SET TIMEOUT STATUS
                          ;
                          ;
                          ;
                          ;
                          ;
        EB10 60           RRETRN  RTS             RETURN
                          ;
                          ;
                          ;
                          ;
                          ;
                          ;SERIAL INPUT READY INTERRUPT SERVICE ROUTINE
                          ;
        EB11 98           ISRSIR  TYA
        EB12 48                   PHA             SAVE Y REG ON STACK
                          ;
                          ;
        EB13 AD 0F D2             LDA     SKSTAT
        EB16 8D 0A D2             STA     SKRES   RESET STATUS REGISTER
                          ;*********  THIS MAY NOT BE THE PLACE TO DO IT   *********
                          ;
        EB19 10 04                BPL     NTFRAM  BRANCH IF NO FRAMING ERROR
```

```
                      ;
E81B A0 8C                    LDY    #FRMERR
E81D 84 30                    STY    STATUS  SET FRAME ERROR STATUS
                      ;
E81F 29 20            NTFRAM  AND    #$20
E821 D0 04                    BNE    NTOVRN  BRANCH IF NO OVERRUN ERROR
                      ;
E823 A0 8E                    LDY    #OVRRUN
E825 84 30                    STY    STATUS  SET OVERRUN ERROR STATUS
                      ;
E827 A5 38            NTOVRN  LDA    BUFRFL
E829 F0 13                    BEQ    NOTYET  BRANCH IF BUFFER WAS NOT YET FILLED
                      ;
E82B AD 0D D2                 LDA    SERIN   THIS INPUT BYTE IS THE CHECKSUM
E82E C5 31                    CMP    CHKSUM
E830 F0 04                    BEQ    SRETRN  BRANCH IF CHECKSUMS MATCH
                      ;
E832 A0 8F                    LDY    #CHKERR
E834 84 30                    STY    STATUS  SET CHECKSUM ERROR STATUS
                      ;
E836 A9 FF            SRETRN  LDA    #$FF    SET RECEIVE DONE FLAG
E838 85 39                    STA    RECVDN
                      ;
E83A 68               SUSUAL  PLA
E83B A8                       TAY            RESTORE Y REG
E83C 68                       PLA            RETURN FROM INTERRUPT
E83D 40                       RTI
                      ;
                      ;
                      ;
E83E AD 0D D2         NOTYET  LDA    SERIN
E841 A0 00                    LDY    #0
E843 91 32                    STA    (BUFRLO),Y      STORE INPUT REGISTER INTO BUFFER
                      ;
E845 18                       CLC            AND IT TO CHECKSUM
E846 65 31                    ADC    CHKSUM
E848 69 00                    ADC    #0
E84A 85 31                    STA    CHKSUM
                      ;
E84C E6 32                    INC    BUFRLO  INCREMENT BUFFER POINTER
E84E D0 02                    BNE    NTWRP1
E850 E6 33                    INC    BUFRHI
                      ;
E852 A5 33            NTWRP1  LDA    BUFRHI
E854 C5 35                    CMP    BFENHI
E856 90 E2                    BCC    SUSUAL
E858 A5 32                    LDA    BUFRLO
E85A C5 34                    CMP    BFENLO
E85C 90 DC                    BCC    SUSUAL  BRANCH IF NEW BUFFER ADDRESS IS IN BUFFER LIMITS
                      ;
E85E A5 3C                    LDA    NOCKSM
E860 F0 06                    BEQ    GOON    BRANCH IF A CHECKSUM WILL FOLLOW DATA
                      ;
E862 A9 00                    LDA    #0
E864 85 3C                    STA    NOCKSM  CLEAR NO CHECKSUM FLAG
                      ;
E866 F0 CE                    BEQ    SRETRN  GO RETURN AND SET RECEIVE DONE FLAG
                      ;
                      ;
E868 A9 FF            GOON    LDA    #$FF
```

```
EB6A 85 38                  STA     BUFRFL  SET BUFFER FULL FLAG
EB6C D0 [illegible]                  BNE     SUSUAL  GO RETURN
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;LOAD BUFFER POINTER SUBROUTINE
                     ;
                     ;LOAD BUFFER POINTER WITH DCB BUFFER INFORMATION
                     ;
EB6E 18              LDPNTR CLC
EB6F AD 04 03               LDA     DBUFLO
EB72 85 32                  STA     BUFRLO
EB74 6D 08 03               ADC     DBYTLO
EB77 85 34                  STA     BFENLO  ALSO SET BUFFER END + 1 ADDRESS
                     ;
EB79 AD 05 03               LDA     DBUFHI
EB7C 85 33                  STA     BUFRHI
EB7E 6D 09 03               ADC     DBYTHI
EB81 85 35                  STA     BFENHI
                     ;
EB83 60                     RTS             RETURN
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;CASSETTE HANDLING CODE
                     ;
EB84 AD 03 03        CASENT LDA     DSTATS
EB87 10 26                  BPL     CASRED  BRANCH IF INPUT FROM CASSETTE
                     ;
                     ;WRITE A RECORD
                     ;
EB89 A9 CC                  LDA     #B00600 SET BAUD RATE TO 600
EB8B 8D 04 D2               STA     AUDF3
EB8E A9 05                  LDA     #B006HI
EB90 8D 06 D2               STA     AUDF4
                     ;
EB93 20 [illegible]                  JSR     SENDEN  TURN ON POKEY MARK TONE
                     ;
EB96 A0 [illegible]                  LDY     #WSIRGLO LOAD SHORT WRITE INTER RECORD GAP TIME
EB98 AD 0B 03               LDA     DAUX2
EB9B 30 02                  BMI     SRTIR0  BRANCH IF SHORT GAP IS DESIRED
                     ;
EB9D A0 [illegible]                  LDY     #WIRGLO SET WRITE IRG TIME
EB9F A2 [illegible]          SRTIR0 LDX     #WIRGHI
EBA1 20 5C E4               JSR     SETVBX
                     ;
EBA4 A9 34                  LDA     #MOTRGO
EBA6 8D 02 D3               STA     PACTL   TURN ON MOTOR
```

```
EDA9 AD 17 03        TIMIT   LDA    TIMFLG  LOOP UNTIL DONE
EDAC D0 FB                   BNE    TIMIT
                     ;
EDAE 20 8E ED                JSR    LDPNTR  LOAD BUFFER POINTER WITH DCB INFORMATION
                     ;
EDB1 20 [illegible]          JSR    SEND    SEND A BUFFER
                     ;
EDB4 4C E3 ED                JMP    CRETRN  GO RETURN
                     ;
                     ;
                     ;READ A RECORD
                     ;
EDB7 A9 FF           CASRED  LDA    #$FF
EDB9 8D 0F 03                STA    CASFLG  SET SET CASSETTE FLAG
                     ;
EDBC A0 0A                   LDY    #RSIRG  LOAD SHORT READ INTER RECORD GAP TIME
EDBE AD 0B 03                LDA    DAUX2
EDC1 30 02                   BMI    SRTIR1  BRANCH IF SHORT GAP IS DESIRED
                     ;
EDC3 A0 78                   LDY    #RIRGLO SET TIME OUT FOR READ IRG
EDC5 A2 00           SRTIR1  LDX    #RIRGHI
EDC7 20 D8 ED                JSR    SETVBX
                     ;
EDCA A9 34                   LDA    #MOTRGO
EDCC 8D 02 D3                STA    PACTL   TURN ON MOTOR
                     ;
EDCF AD 17 03        TIMIT1  LDA    TIMFLG  LOOP UNTIL DONE
EDD2 D0 FB                   BNE    TIMIT1
                     ;
EDD4 20 8E ED                JSR    LDPNTR  LOAD BUFFER POINTER WITH DCB INFORMATION
                     ;
EDD7 20 79 EC                JSR    STTMOT  SET DEVICE TIME OUT IN Y,X
EDDA 20 D8 ED                JSR    SETVBX
                     ;
EDDD 20 34 E9                JSR    BEGIN   SET INITIAL BAUD RATE
                     ;
EDE0 20 EA E9                JSR    RECEIV  GO RECEIVE A BLOCK
                     ;
EDE3 AD 0B 03        CRETRN  LDA    DAUX2
EDE6 30 05                   BMI    SRTIR2  BRANCH IF DOING SHORT INTER RECORD GAPS
                                            DON'T TURN OFF CASSETTE MOTOR
                     ;
EDE8 A9 3C                   LDA    #MOTRST
EDEA 8D 02 D3                STA    PACTL   TURN OFF MOTOR
                     ;
EDED 4C 5D E9        SRTIR2  JMP    RETURN  GO RETURN
                     ;
                     ;
                     ;
                     ;
EDF0 A9 00           JTIMER  LDA    #$00
EDF2 8D 17 03                STA    TIMFLG  SET TIME OUT FLAG
EDF5 60                      RTS
                     ;
                     ;
                     ;
                     ;
                     ;
                     ;
```

```
SIO ( SERIAL BUS INPUT/OUTPUT CONTROLLER )

                        ;
                        ;SEND ENABLE SUBROUTINE
                        ;
EBF6 A9 07              SENDEN  LDA   #$07    MASK OFF PREVIOUS SERIAL BUS CONTROL BITS
EBF8 2D 32 02                   AND   SSKCTL
EBFB 09 20                      ORA   #$20    SET TRANSMIT MODE

EBFD AC 00 03                   LDY   DDEVIC
EC00 C0 60                      CPY   #CASSET
EC02 D0 0C                      BNE   NOTCAS  BRANCH IF NOT CASSETTE
                        ;
EC04 09 08                      ORA   #$08    SET THE FSK OUTPUT BIT
                        ;
EC06 A0 07                      LDY   #LOTONE SET FSK TONE FREQUENCIES
EC08 8C 02 D2                   STY   AUDF2
EC0B A0 05                      LDY   #HITONE
EC0D 8C 00 D2                   STY   AUDF1
                        ;
EC10 8D 32 02           NOTCAS  STA   SSKCTL  STORE NEW VALUE TO SYSTEM MASK
EC13 8D 0F D2                   STA   SKCTL   STORE TO ACTUAL REGISTER

EC16 A9 C7                      LDA   #$C7    MASK OFF PREVIOUS SERIAL BUS INTERRUPT BITS
EC18 25 10                      AND   POKMSK
EC1A 09 10                      ORA   #$10    ENABLE OUTPUT DATA NEEDED INTERRUPT
                        ;
                        ;
EC1C 4C 35 EC                   JMP   CONTIN  GO CONTINUE IN RECEIVE ENABLE SUBROUTINE
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;RECEIVE ENABLE SUBROUTINE
                        ;
EC1F A9 07              RECVEN  LDA   #$07    MASK OFF PREVIOUS SERIAL BUS CONTROL BITS
EC21 2D 32 02                   AND   SSKCTL
EC24 09 10                      ORA   #$10    SET RECEIVE MODE ASYNCH.
EC26 8D 32 02                   STA   SSKCTL  STORE NEW VALUE TO SYSTEM MASK
EC29 8D 0F D2                   STA   SKCTL   STORE TO ACTUAL REGISTER
                        ;
EC2C 8D 0A D2                   STA   SKRES   RESET SERIAL PORT/KEYBOARD STATUS REGISTER
                        ;
EC2F A9 C7                      LDA   #$C7    MASK OFF PREVIOUS SERIAL BUS INTERRUPT BITS
EC31 25 10                      AND   POKMSK
EC33 09 20                      ORA   #$20    ENABLE RECEIVE INTERRUPT
EC35 85 10              CONTIN  STA   POKMSK  STORE NEW VALUE TO SYSTEM MASK
EC37 8D 0E D2                   STA   IRQEN   STORE TO ACTUAL REGISTER
                        ;
                        ;
EC3A A9 28                      LDA   #$28    CLOCK CH.3 WITH 1.79 MHZ
EC3C 8D 08 D2                   STA   AUDCTL  CLOCK CH.4 WITH CH. 3

EC3F A2 06                      LDX   #6      SET PURE TONES, NO VOLUME
EC41 A9 A8                      LDA   #$A8
EC43 A4 41                      LDY   SOUNDR  TEST QUIET I/O FLAG
EC45 D0 02                      BNE   CHANGE  NE IS NORMAL (NOISY)
```

```
      EC47 A9 A8                   LDA    #$A8
      EC49 9D 01 D2       NOISE1   STA    AUDC1,X
      EC4C CA                      DEX
      EC4D CA                      DEX
      EC4E 10 F9                   BPL    NOISE1

      EC50 A9 A0                   LDA    #$A0
      EC52 8D 05 D2                STA    AUDC3    TURN OFF SOUND ON CHANNEL 3
      EC55 AC 00 03                LDY    DDEVIC
      EC58 C0 60                   CPY    #CASET
      EC5A F0 06                   BEQ    CASSI    BRANCH IF CASSETTE IS DESIRED
      EC5C 8D 01 D2                STA    AUDC1    OTHERWISE TURN OFF CHANNELS 1 AND 2
      EC5F 8D 03 D2                STA    AUDC2

      EC62 60             CASSI    RTS                       RETURN




                         ;DISABLE SEND AND DISABLE RECEIVE SUBROUTINES
                         ;
      EC63 EA             SENDDS   NOP
      EC64 A9 C7          RECVDS   LDA    #$C7     MASK OFF SERIAL BUS INTERRUPTS
      EC66 25 10                   AND    POKMSK
      EC68 85 10                   STA    POKMSK   STORE NEW VALUE TO SYSTEM MASK
      EC6A 8D 0E D2                STA    IRQEN    STORE TO ACTUAL REGISTER

      EC6D A2 06                   LDX    #6
      EC6F A9 00                   LDA    #0
      EC71 9D 01 D2       ZERIT    STA    AUDC1,X
      EC74 CA                      DEX
      EC75 CA                      DEX
      EC76 10 F9                   BPL    ZERIT            TURN OFF AUDIO VOLUME

      EC78 60                      RTS                     RETURN




                         ;GET DEVICE TIME OUT VALUES IN Y,X SUBROUTINE

      EC79 AD 06 03       STTMOT   LDA    DTIMLO   GET DEVICE TIME OUT IN 1 SECOND INCR
      EC7C 6A                      ROR    A        PUT 6 HI BITS IN X, LO 2 BITS IN Y
      EC7D 6A                      ROR    A
      EC7E A8                      TAY             TEMP SAVE
      EC7F 29 3F                   AND    #$3F     MASK OFF 2 HI BITS
      EC81 AA                      TAX             THIS IS HI BYTE OF TIME OUT
```

```
        EC7F 29 3F                      AND     #$3F     MASK OFF 2 HI BITS
        EC81 A8                         TAY              THIS IS HI BYTE OF TIME OUT

SIO ( SERIAL BUS INPUT/OUTPUT CONTROLLER )

        EC82 68                         PLA              RESTORE
        EC83 0A                         ASL     A
        EC84 29 C0                      AND     #$C0     MASK OFF ALL BUT 2 HI BITS
        EC86 AA                         TAX              THIS IS LO BYTE OF TIME OUT
                                ;
        EC87 60                         RTS
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
                                ;
        EC88 11 EB              TOFTVB  WORD    ISRSIR  SERIAL INPUT READY
        EC8A 96 EA                      WORD    ISRODN  OUTPUT DATA NEEDED
        EC8C D1 EA                      WORD    ISRTD   TRANSMISSION DONE
                                ;
        00EB                    SIRHI   =   ISRSIR/256   SERIAL INPUT READY ISR ADDRESS
        0011                    SIRLO   =   -256*SIRHI+ISRSIR
        00EA                    ODNHI   =   ISRODN/256   OUTPUT DATA NEEDED ISR ADDRESS
        0096                    ODNLO   =   -256*ODNHI+ISRODN
        00EA                    TDHI    =   ISRTD/256    TRANSMISSION DONE ISR ADDRESS
        00D1                    TDLO    =   -256*TDHI+ISRTD
                                ;
                                ;
                                ;
                                ;SEND A DATA FRAME TO AN INTELLIGENT PERIPHERAL SUBROUTINE
                                ;
                                ;
        EC8E A2 01              SENDIN  LDX     #$01
        EC90 A0 FF              DELAY0  LDY     #$FF
        EC92 88                 DELAY1  DEY
        EC93 D0 FD                      BNE     DELAY1
        EC95 CA                         DEX
        EC96 D0 F8                      BNE     DELAY0
                                ;
        EC98 20 A8 EA                   JSR     SEND     GO SEND THE DATA FRAME
                                ;
        EC9B A0 02                      LDY     #CTIMLO SET ACK TIME OUT
        EC9D A2 00                      LDX     #CTIMHI
        EC9F 20 B0 ED           WAITER  JSR     SETVBX
                                ;
        ECA2 20 1A EA                   JSR     WAIT     WAIT FOR ACK
                                ;
        ECA5 98                         TYA              IF Y<0, A TIME OUT OR NACK OCCURED
                                ;
        ECA6 60                         RTS              RETURN
                                ;
                                ;
                                ;
                                ;
                                ;
```

```
                          ;
                          ;
                          ;        COMPUTE VALUE FOR POKEY FREQ REGS FOR THE BAUD RATE AS
                          ;                MEASURED BY AN INTERVAL OF THE 'VCOUNT' TIMER.
                          ;
ECA7 8D 10 03      COMPUT  STA     TIMER2
ECAA 8C 11 03              STY     TIMER2+1          SAVE FINAL TIMER VALUE
ECAD 20 08 ED              JSR     ADJUST            ADJUST VCOUNT VALUE
ECB0 8D 10 03              STA     TIMER2            SAVE ADJUSTED VALUE
ECB3 AD 0C 03              LDA     TIMER1
ECB6 20 08 ED              JSR     ADJUST            ADJUST
ECB9 8D 0C 03              STA     TIMER1            SAVE ADJUSTED TIMER1 VALUE
ECBC AD 10 03              LDA     TIMER2
ECBF 38                    SEC
ECC0 ED 0C 03              SBC     TIMER1
ECC3 8D 12 03              STA     TEMP1             FIND VCOUNT DIFFERENCE
ECC6 AD 11 03              LDA     TIMER2+1
ECC9 38                    SEC
ECCA ED 0D 03              SBC     TIMER1+1
ECCD A8                    TAY                       FIND VBLANK COUNT DIFFERENCE
ECCE A9 7D                 LDA     #-$83
ECD0 18            HITIMR  CLC
ECD1 69 83                 ADC     #$83              ACCUMULATE MULTIPLICATION
ECD3 88                    DEY
ECD4 10 FA                 BPL     HITIMR            DONE?
ECD6 18                    CLC
ECD7 6D 12 03              ADC     TEMP1             TOTAL VCOUNT DIFFERENCE
ECDA A8            FINDX   TAY                       SAVE ACCUM
ECDB 4A                    LSR     A
ECDC 4A                    LSR     A
ECDD 4A                    LSR     A
ECDE 0A                    ASL     A
ECDF 38                    SEC
ECE0 E9 16                 SBC     #22               ADJUST TABLE INDEX
ECE2 AA                    TAX                       DIVIDE INTERVAL BY 4 TO GET TABLE INDEX
ECE3 98                    TYA                       RESTORE ACCUM
ECE4 29 07                 AND     #7
ECE6 A8                    TAY                       PULL OFF 3 LO BITS OF INTERVAL
ECE7 A9 F5                 LDA     #-11
ECE9 18            DOINTP  CLC
ECEA 69 0B                 ADC     #11               ACCUMULATE INTERPOLATION CONSTANT
ECEC 88                    DEY
ECED 10 FA                 BPL     DOINTP            INTERPOLATION CONSTANT COMPUTATION DONE?
                          ;
ECEF A0 00         ENINTP  LDY     #0
ECF1 8C 0E 03              STY     ADDCOR            CLEAR ADDITION CORRECTION FLAG
ECF4 38                    SEC
ECF5 E9 07                 SBC     #7                ADJUST INTERPOLATION CONSTANT
ECF7 10 03                 BPL     PLUS
ECF9 CE 0E 03              DEC     ADDCOR
ECFC 18            PLUS    CLC
ECFD 7D 14 ED              ADC     POKTAB,X          ADD CONSTANT TO LO BYTE TABLE VALUE
ED00 A8                    TAY                       LO BYTE POKEY FREQ VALUE
ED01 AD 0E 03              LDA     ADDCOR
ED04 7D 15 ED              ADC     POKTAB+1,X        ADD CARRY TO HI BYTE TABLE VALUE
                          ;                          HI BYTE POKEY FREQ VALUE
ED07 60                    RTS
                          ;
                          ;
```

```
SIO  ( SERIAL BUS INPUT/OUTPUT CONTROLLER )

                      ;
                      ;       ROUTINE TO ADJUST VCOUNT VALUE          (why?)
                      ;
    ED08 C9 7C        ADJUST  CMP     #$7C
    ED0A 30 04                BMI     ADJ1             LARGER THAN '7C' ?
    ED0C 38                   SEC                      YES,
    ED0D E9 7C                SBC     #$7C
    ED0F 60                   RTS
    ED10 18           ADJ1    CLC
    ED11 69 07                ADC     #7
    ED13 60                   RTS
                      ;
                      ;
                      ;
                      ;
                      ;
                      ;
                      ;       INITIAL BAUD RATE MEASUREMENT -- USED TO SET THE
                      ;               BAUD RATE AT THE START OF A RECORD.
                      ;
                      ;               IT IS ASSUMED THAT THE FIRST TWO BYTES OF EVERY
                      ;               RECORD ARE 'AA' HEX.
                      ;
    ED14 A5 11        BEGIN   LDA     BRKKEY
    ED16 D0 03                BNE     NTBRK2
    ED18 4C A[illegible] E[illegible]              JMP     BROKE    JUMP IF BREAK KEY PRESSED
                      ;
    ED1B 78           NTBRK2  SEI
                      ;
    ED1C AD 17 03             LDA     TIMFLG
    ED1F D0 02                BNE     OKTIM1   BRANCH IF NOT TIMED OUT
    ED21 F0 25                BEQ     TOUT1    BRANCH IF TIME OUT
                      ;
    ED23 AD 0F D2     OKTIM1  LDA     SKSTAT
    ED26 29 10                AND     #$10             READ SERIAL PORT
    ED28 D0 EA                BNE     BEGIN            START BIT?
    ED2A 8D 1E 03             STA     SAVIO            SAVE SER. DATA IN
    ED2D AE 0B D4             LDX     VCOUNT           READ VERTICAL LINE COUNTER
    ED30 A4 14                LDY     RTCLOK+2         READ LO BYTE OF VBLANK CLOCK
    ED32 8E 0C 03             STX     TIMER1
    ED35 8C 0D 03             STY     TIMER1+1         SAVE INITIAL TIMER VALUE
                      ;
    ED38 A2 01                LDX     #1               SET MODE FLAG
    ED3A 8E 15 03             STX     TEMP3
    ED3D A0 0A                LDY     #10              SET BIT COUNTER FOR 10 BITS
    ED3F A5 11        COUNT   LDA     BRKKEY
    ED41 F0 [illegible]                BEQ     BROKE    BRANCH IF BREAK KEY PRESSED
                      ;
    ED43 AD 17 03             LDA     TIMFLG
    ED46 D0 04                BNE     OKTIMR   BRANCH IF NOT TIMED OUT
    ED48 58           TOUT1   CLI
    ED49 4C [illegible] E[illegible]              JMP     TOUT     BRANCH IF TIME OUT
                      ;
    ED4C AD 0F D2     OKTIMR  LDA     SKSTAT
    ED4F 29 10                AND     #$10             READ SERIAL PORT
    ED51 CD 1E 03             CMP     SAVIO            DATA IN CHANGED YET?
    ED54 F0 E9                BEQ     COUNT
    ED56 8D 1E 03             STA     SAVIO            YES,SAVE SER. DATA IN
    ED59 88                   DEY                      DECR. BIT COUNTER
```

```
   [illegible]                    [illegible]  [illegible]    [illegible]
   [illegible]                    INC      TIMR3       YES,
   [illegible]                    BNE      [illegible]READ  DONE WITH BOTH MODES?
   E963 AD [illegible]            LDA      ?COUNT
   E966 A4 14                     LDY      RTCLOK+2    READ TIMER LO & HI BYTES
   E968 20 [illegible]            JSR      COMPUT      NO, COMPUTE BAUD RATE
   E96B 8E EE 02                  STX      CBAUDL
   E96E 8C EF 02                  STY      CBAUDH      SET BAUD RATE INTO RAM CELLS
   E971 A0 09                     LDY      #9          SET BIT COUNTER FOR 9 BITS
   E973 D0 [illegible]            BNE      COUNT

   E975 AD EE 02        RESRAD:   LDA      CBAUDL
   E978 8D 04 D2                  STA      AUDF3
   E97B AD EF 02                  LDA      CBAUDH
   E97E 8D 06 D2                  STA      AUDF4       SET POKEY FREQ REGS FOR BAUD RATE
   E981 A9 00                     LDA      #0
   E983 8D 0F D2                  STA      SKSTAT
   E986 AD 32 02                  LDA      SSKCTL
   E989 8D 0F D2                  STA      SKSTAT      INIT. POKEY SERIAL PORT
   E98C A9 55                     LDA      #$55
   E98E 91 32                     STA      (BUFRLO),Y  STORE '$55' AS FIRST RCV. BUFFER
   E990 C8                        INY
   E991 91 32                     STA      (BUFRLO),Y
   E993 A9 AA                     LDA      #$AA
   E995 85 31                     STA      CHKSUM      STORE CHECKSUM FOR 2 BYTES OF '$AA'
   E997 18                        CLC
   E998 A5 32                     LDA      BUFRLO
   E99A 69 02                     ADC      #2
   E99C 85 32                     STA      BUFRLO
   E99E A5 33                     LDA      BUFRHI
   E9A0 69 00                     ADC      #0
   E9A2 85 33                     STA      BUFRHI      INCR. BUFFER POINTER BY 1
   E9A4 58                        CLI
   E9A5 60                        RTS
                        :
                        :
                        :
   E9A6 20 63 EC        BROKE:    JSR      SENDDS   BREAK KEY WAS PRESSED, SO PREPARE
   E9A9 A9 3C                     LDA      #MOTRST  TO RETURN
   E9AB 8D 02 D3                  STA      PACTL    TURN OFF MOTOR
   E9AE 8D 03 D3                  STA      PBCTL    RAISE NOT COMMAND LINE
                        :
   E9B1 A9 80                     LDA      #BRKABT
   E9B3 85 30                     STA      STATUS   STORE BREAK ABORT STATUS CODE
                        :
   E9B5 AE 18 03                  LDX      STACKP
   E9B8 9A                        TXS               RESTORE STACK POINTER

   E9B9 C6 11                     DEC      BRKKEY   SET BREAK KEY FLAG TO NONZERO
   E9BB 58                        CLI               ALLOW IRQ'S
                        :
   E9BC 4C [illegible]            JMP      RETURN   GO RETURN
                        :
                        :
                        :
                        :
   E9BF A9 EC           SETVBX:   LDA      #<TIMOUT STORE TIME OUT ROUTINE ADDRESS
   E9C1 8D 26 02                  STA      CDTMA1
```

```
EDBF 8D 26 02                STA    CDTMA1

EDC2 A9 EB                   LDA    #JTADRH
EDC4 8D 27 02                STA    CDTMA1+1
                  ;
EDC7 A9 01                   LDA    #1        ;SET FOR TIMER 1
EDC9 8D 17 03                STA    TIMFLG    ;SET FLAG TO NOT TIMED OUT
                  ;
EDCC 78                      SEI              ;THE SETVBL ROUTINE NEEDS THIS TO CUT SHORT
EDCD 20 5C E4                JSR    SETVBV    ;ANY VBLANKS THAT OCCUR
EDD0 58                      CLI
EDD1 60                      RTS
                  ;
                  ;
                  ;
                  ;
                  ;
                  ;
                  ;          'VCOUNT' INTERVAL TIMER MEASUREMENT -- TO -- POKEY FREQ REG VALUE
                  ;                   CONVERSION TABLE
                  ;
                  ;                   THE VALUES STORED IN THE TABLE ARE 'AUDF+7'.
                  ;
                  ;          THE FOLLOWING FORMULAS WERE USED TO DETERMINE THE TABLE VALUES:
                  ;
                  ;                   F OUT= F IN/(2*(AUDF+M)) , WHERE F IN=1.78979 MHZ. & M=7
                  ;
                  ;                   FROM THIS WAS DERIVED THE FORMULA USED THE COMPUTE THE
                  ;                     TABLE VALUES BASED ON A MEASUREMENT OF THE PERIOD BY
                  ;                     AN INTERVAL OF THE 'VCOUNT' TIMER.
                  ;
                  ;                   AUDF+7=(11.365167)*T OUT, WHERE T OUT=# OF COUNTS (127 USEC.
                  ;                                                  RESOLUTION) OF 'VCOUNT' FOR 1
                  ;                                                  CHARACTER TIME (10 BIT TIMES).
                  ;
                  ;
                  ;
                  ;                   AUDF+7            BAUD RATE          VCOUNT INTERVAL
                  ;                   ------            ---------          ---------------
                  ;          .WORD    $27C              ;1407                    56
                  ;          .WORD    $2D7              ;1231                    64
                  ;          .WORD    $332              ;1094                    72
                  ;          .WORD    $38D              ;985                     80
EDD2 E8 03        POKTAB     .WORD    $3E8              ;895                     88
EDD4 43 04                   .WORD    $443              ;820                     96
EDD6 9E 04                   .WORD    $49E              ;757                     104
EDD8 F9 04                   .WORD    $4F9              ;703                     112
EDDA 54 05                   .WORD    $554              ;656                     120
EDDC AF 05                   .WORD    $5AF              ;615                     128
EDDE 0A 06                   .WORD    $60A              ;579                     136
EDE0 65 06                   .WORD    $665              ;547                     144
EDE2 C0 06                   .WORD    $6C0              ;518                     152
EDE4 1A 07                   .WORD    $71A              ;492                     160
EDE6 75 07                   .WORD    $775              ;469                     168
EDE8 D0 07                   .WORD    $7D0              ;447                     176
                  ;          .WORD    $828              ;428                     184
                  ;          .WORD    $886              ;410                     192
                  ;          .WORD    $8E1              ;394                     200
                  ;          .WORD    $93C              ;379                     208
```

```
                        ;       .WORD   $997            ;365                    216
                        ;       .WORD   $9F2            ;352                    224
                        ;       .WORD   $A4D            ;339                    230
                        ;       .WORD   $AAB            ;328                    240
                        ;       .WORD   $B03            ;318                    248
                        ;
                        ;
                        ;
                        ;
                        ;***********************************************************************
     E0EA               CRNTPC  =*
                                *=$14
     0014 00            SIOSPR  .BYTE   DSKORG-CRNTPC   SIOL IS TOO LONG
                        .END
```

SIO (SERIAL BUS INPUT/OUTPUT CONTROLLER)

```
              SYMBOL  TABLE
  ACK       0041     [illegible] [illegible]   [illegible] 0306     [illegible] ED10
  ADJUST    E000     ADRESS    0069       AFP       D800       [illegible] 0203
  ANTIC     D400     APPEND    0001       APPMHI    000E       ATACHR    02FB
  ATAN      [illegible] ATRACT 004D       AUDC1     D201       AUDC2     D203
  AUDC3     D205     AUDC4     D207       AUDCTL    D208       AUDF1     D200
  AUDF2     D202     AUDF3     D204       AUDF4     D206       [illegible] 0006
  B192LO    0028     [illegible] [illegible]   BAUDLO    00CC       [illegible] EA63
  [illegible] [illegible] [illegible] 0086    [illegible] 0091     [illegible] ED1A
  [illegible] [illegible] [illegible] [illegible] [illegible] 0065 [illegible] 02BA
  [illegible] E471   [illegible] [illegible]   [illegible] 0242     BOTSCR    02BF
  BPTR      003D     [illegible] [illegible]   BRKKEY    0011       [illegible] [illegible]
  BUFADR    0015     BUFCNT    006B       BUFRFL    0038       BUFRHI    0033
  BUFRLO    0032     BUFSTR    006C       [illegible] EC62     CASBUF    03FD
  CASENT    [illegible] CASETV E440       CASFLG    030F       CASINI    0002
  [illegible] EF41   [illegible] [illegible]   [illegible] 0048     [illegible] 0050
  CAUX1     023C     CAUX2     023D       CBAUDH    02EF       CBAUDL    02EE
  [illegible] [illegible] [illegible] [illegible] CCOMND 023B    CDEVIC    023A
  CDTMA1    0226     CDTMA2    0228       CDTMF3    022A       CDTMF4    022C
  CDTMF5    022E     CDTMV1    0218       CDTMV2    021A       CDTMV3    021C
  CDTMV4    021E     CDTMV5    0220       CH        02FC       CH1       02F2
  CHACT     02F3     CHACTL    D401       CHAR      02FA       CHBAS     02F4
  CHBASE    D409     [illegible] [illegible]   [illegible] 008E     CHKSNT    003B
  CHKSUM    0031     [illegible] [illegible]   CHRORG    E000       CIOCHR    002F
  [illegible] [illegible] [illegible] [illegible] CIOV   E456     CIX       00F2
  CKEY      004A     CLOSE     000C       COLAC     0072       COLBK     D01A
  COLCRS    0055     COLDST    0244       COLDSV    E477       COLINC    007A
  COLOR0    02C4     COLOR1    02C5       COLOR2    02C6       COLOR3    02C7
  COLOR4    02C8     COLPF0    D016       COLPF1    D017       COLPF2    D018
  COLPF3    D019     COLPM0    D012       COLPM1    D013       COLPM2    D014
  COLPM3    D015     COLRSH    004F       [illegible] [illegible] [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] CONSOL D01F   [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] COUNTR 007E   CR        009B
  [illegible] [illegible] [illegible] [illegible] CRETRY 0036   CRITIC    0042
  [illegible] [illegible] CRSINH 02F0     [illegible] [illegible] [illegible] E470
  [illegible] [illegible] CTIA   D000     [illegible] 0000     [illegible] 0002
  DAUX1     030A     DAUX2     030B       [illegible] 0041     DBUFHI    0305
  DBUFLO    0304     DBYTHI    0309       DBYTLO    0308       DCB       0300
  DCOMND    0302     DDEVIC    0300       [illegible] 00FB     [illegible] 0006
  [illegible] [illegible] [illegible] [illegible] [illegible] 0021 DELTAC 0077
  DELTAR    0076     [illegible] [illegible]   [illegible] EA06     [illegible] 0090
  [illegible] [illegible] [illegible] [illegible] [illegible] 0007 DIRECT 0002
  [illegible] [illegible] [illegible] [illegible] [illegible] 0050 DLISTH D403
  DLISTL    D402     DMACTL    D400       [illegible] 02A0     [illegible] [illegible]
  [illegible] [illegible] DOSINI 000C     DOSVEC    000A       [illegible] 0044
  [illegible] [illegible] DRETRI 0037     [illegible] 0037     [illegible] 004E
  [illegible] [illegible] DSKINV E453     [illegible] EDEA     [illegible] 0236
  [illegible] [illegible] [illegible] [illegible] DSTAT  004C   DSTATS    0303
  [illegible] [illegible] DUNIT  0301     [illegible] [illegible] DVSTAT  02EA
  EDITRV    E400     [illegible] [illegible]   [illegible] 0074     [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] [illegible] 0045 [illegible] 02A2
  ESIGN     00EF     [illegible] [illegible]   [illegible] 00CC     [illegible] DAA6
  FASC      D8E6     FCHRFL    00F0       [illegible] 0025     [illegible] 00CF
  FILDAT    02FD     [illegible] [illegible]   [illegible] 0012     [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] [illegible] 003C [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] [illegible] 0063
  [illegible] [illegible] [illegible] [illegible] [illegible] DAE0 [illegible] 0021
  FPI       D9D2     [illegible] [illegible]   [illegible] D9EA     [illegible] [illegible]
  [illegible] [illegible] [illegible] [illegible] [illegible] 00E0 [illegible] 00E6
  [illegible] [illegible] [illegible] [illegible] [illegible] 00FC [illegible] 00FC
```

```
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  00A6  [illegible]  00A3
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  0007  [illegible]  0005
GLBABS  02E0    [illegible]  [illegible]  [illegible]  EA53  [illegible]  ED6B
[illegible]  [illegible]  GPRIOR  026F    GRACTL  D01D    GRAFM   D011
GRAFP0  D00D    GRAFP1  D00E    GRAFP2  D00F    GRAFP3  D010
[illegible]  [illegible]  HITCLR  D01E    [illegible]  EDD0  [illegible]  0005
HOLD1   0051    HOLD2   029F    HOLD3   029D    HOLD4   02BE
[illegible]  [illegible]  HOLDCH  [illegible]  HPOSM0  D004    HPOSM1  D005
HPOSM2  D006    HPOSM3  D007    HPOSP0  D000    HPOSP1  D001
HPOSP2  D002    HPOSP3  D003    HSCROL  D404    ICAX1   034A
ICAX1Z  002A    ICAX2   034B    ICAX2Z  002B    ICBAH   0345
ICBAHZ  0025    ICBAL   0344    ICBALZ  0024    ICBLH   0349
ICBLHZ  0029    ICBLL   0348    ICBLLZ  0028    ICCOM   0342
ICCOMT  0017    ICCOMZ  0022    ICDNO   0341    ICDNOZ  0021
ICHID   0340    ICHIDZ  0020    ICIDNO  002E    ICPTH   0347
ICPTHZ  0027    ICPTL   0346    ICPTLZ  0026    ICSPR   034C
ICSPRZ  002C    ICSTA   0343    ICSTAZ  0023    IFP     D9AA
INBUFF  00F3    [illegible]  [illegible]  INSDAT  007D    INTABS  0200
INTEMP  022D    [illegible]  [illegible]  INTORG  EAD5    [illegible]  ECA9
[illegible]  0010    INVFLG  02B6    IOCB    0340    IOCBAS  0020
IOCBSZ  0010    IOCFRE  00FF    IRQEN   D20E    IRQST   D20E
ISRODN  EA90    ISRSIR  EB11    ISRTD   EAD1    JTADRH  00FB
JTADRL  [illegible]  JTIMER  [illegible]  KBCODE  D209    [illegible]  0048
[illegible]  [illegible]  KEYBDV  E420    KEYDEL  02F1    [illegible]  00FF
LBPR1   057E    LBPR2   057F    LBUFF   0580    [illegible]  E96E
LEDGE   0002    LINBUF  0247    LINZBS  0000    LMARGN  0052
[illegible]  0011    [illegible]  [illegible]  LOG10   DED1    LOGCOL  0063
LOGMAP  02B2    [illegible]  0007    LPENH   0234    LPENV   0235
M0PF    D000    M0PL    D008    M1PF    D001    M1PL    D009
M2PF    D002    M2PL    D00A    M3PF    D003    M3PL    D00B
MAXDEV  0021    MAXIOC  0080    MEMLO   02E7    MEMTOP  02E5
MLTTMP  0066    [illegible]  [illegible]  MODEM   0040    [illegible]  [illegible]
[illegible]  0034    [illegible]  003C    [illegible]  0010    [illegible]  00AE
[illegible]  [illegible]  [illegible]  [illegible]  NEWCOL  0061    NEWROW  0060
NMIEN   D40E    NMIRES  D40F    NMIST   D40F    NOCKSM  003C
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  0082    [illegible]  004E
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  EA52    [illegible]  EAB1
[illegible]  [illegible]  [illegible]  EAA0    [illegible]  EAD0    [illegible]  D085
NOTYET  EA0E    [illegible]  [illegible]  [illegible]  00EE    [illegible]  EA88
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  EB1F    NTOVRN  EB27
[illegible]  EB52    [illegible]  [illegible]  [illegible]  EA56    [illegible]  00EA
[illegible]  0096    [illegible]  [illegible]  [illegible]  ED4C    OLDADR  005E
OLDCHR  005D    OLDCOL  005B    OLDROW  005A    OPEN    0003
OPNIN   0004    OPNINO  000C    OPNOT   0008    [illegible]  [illegible]
OVRRUN  008E    P0PF    D004    P0PL    D00C    P1PF    D005
P1PL    D00D    P2PF    D006    P2PL    D00E    P3PF    D007
P3PL    D00F    PACTL   D302    PADDL0  0270    PADDL1  0271
PADDL2  0272    PADDL3  0273    PADDL4  0274    PADDL5  0275
PADDL6  0276    PADDL7  0277    PBCTL   D303    [illegible]  [illegible]
[illegible]  [illegible]  PCOLR0  02C0    PCOLR1  02C1    PCOLR2  02C2
PCOLR3  02C3    [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  D300
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  05E0    [illegible]  [illegible]
[illegible]  [illegible]  [illegible]  [illegible]  POKEY   D200    POKMSK  0010
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  D201    [illegible]  D200
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  D203    [illegible]  D204
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]
[illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  [illegible]  001C
PTRIG0  027C    PTRIG1  027D    PTRIG2  027E    PTRIG3  027F
PTRIG4  0280    PTRIG5  0281    PTRIG6  0282    PTRIG7  0283
```

```
PTRIG0  027C      PTRIG1  027D      PTRIG2  027E      PTRIG3  027F
PTRIG4  0280      PTRIG5  0281      PTRIG6  0282      PTRIG7  0283
```

SIO (SERIAL BUS INPUT/OUTPUT CONTROLLER)

```
PUTCHR  000A      PXITEL  0009      RADFLG  00FB      RADON   0000
RAMLO   0004      RAMSIZ  02E4      RAMTOP  006A      RANDOM  D20A
RBLOKV  E47A      RDONLY  0001      READ    0052      RECEIV  EAE2
RECVDN  0039      RECVDS  EB64      RECVEN  EC1F      RENDE   0027
RELONE  EA03      RENAME  0020      RETURN  EA0D      RIGHHI  0000
RIRGLO  0078      RMARGN  0053      ROWAC   0070      ROWCRS  0054
RUNINC  0079      RRETRN  EB10      RWIRQ   000A      RTCLOK  0012
SAVADR  0068      SAVIO   0316      SAVMSC  0058      SCREAT  0045
SCRENV  E410      SCRFLG  02BB      SCRMEM  0093      SDLSTH  0231
SDLSTL  0230      SDMCTL  022F      SEND    EA88      SENDDS  EC6D
SENDEN  EBE6      SENDEV  E468      SENDIN  EC8E      SERIN   D20D
SEROUT  D20D      SETVBV  E45C      SETVBX  E8BD      SHFAMT  006F
SHFLOK  02BE      SIOWAY  0002      SIN     BD81      SIO     E959
SIOINT  E944      SIOINV  E465      SIOSRC  E944      SIOSPG  0019
SIOV    E459      SIRHI   00FB      SIRLO   0011      SIZEM   000C
SIZEP0  0008      SIZEP1  0009      SIZEP2  000A      SIZEP3  000B
SKCTL   D20F      SKRES   D20A      SKSTAT  D20F      SOUNDR  0041
SPECIL  000E      SQR     BF81      SRETRN  EB36      SRTIMR  022B
SRTIRQ  EB9F      SRTIR1  EB05      SRTIR2  EBED      SSFLAG  02FF
SSKCTL  0232      STACKP  0318      STATIS  0053      STATUS  0030
STICK0  0278      STICK1  0279      STICK2  027A      STICK3  027B
STIMER  D209      STRIG0  0284      STRIG1  0285      STRIG2  0286
STRIG3  0287      STTMOT  EC79      SUBTMP  029E      SUCCES  0001
SUSWAI  E8DA      SWPFLG  007B      SYSVBV  E45F      TABMAP  02A3
TDUT    00FA      TOLD    00D1      TEMP    023E      TEMP1   0312
TEMP2   0314      TEMP3   0315      TERMBT  0302      TESTLN  003E
TIMER1  030C      TIMER2  0310      TIMFLG  0317      TIMIT   EBAY
TIMIT1  EBCF      TIMOUT  D00A      TINDEX  0293      TMPCHR  0050
TMPCOL  02B9      TMPLBT  02A1      TMPROW  02B8      TMPX1   029C
TOUT    E00C      TOUT1   E990      TRAMSZ  0006      TRKBD   0010
TRIG1   D011      TRIG2   D012      TRIG3   D013      TRNRED  0087
TSSTAT  0011      TSTDAT  0007      TXTCOL  0291      TXTMSC  0294
TXTOLD  0296      TXTROW  0290      UNLOCK  0024      USAREA  0480
VBREAK  0206      VCOUNT  D40B      VCTABL  E480      VDELAY  D01C
VDSLST  0200      VECTBL  E400      VIMIRQ  0216      VINTER  0204
VKEYBD  0208      VPRCED  0202      VSBUOL  0905      VSERIN  020A
VSEROC  020E      VSEROR  020C      VTIMR1  0210      VTIMR2  0212
VTIMR4  0214      VVBLKD  0224      VVBLKI  0222      WAIT    EA1A
WAITER  EC9F      WARMST  0008      WARMSV  E474      WAYCOM  E9D2
WIRGHT  0000      WIRGLO  00B4      WMODE   0289      WSP     EA30
WRITE   0057      WRONLY  0008      WS1BGC  010F      WSYNC   D40A
XITVBV  E462      XMTDON  003A      XMTIT   EC71      ZIOCB   0020
ZTEMP1  00F5      ZTEMP3  00F9      ZTEMP4  00F7
```

```
                        ;
                        .TITLE 'DISK  ***** DISKP.SRC  ***** 3/9/79 ***** 4:00:00 P.M.'
                        ;
                        ;
                        ;
                        ;
        0002            STATVH  =       DVSTAT/256
        00EA            STATVL  =       -256*STATVH+DVSTAT      ;STATUS POINTER
                        ;
                        ;
                        ;
                        ;
                        ;       CONSTANT EQUATES
                        ;
        0031            DISKID  =       $31             ;SERIAL BUS DISK I.D.
        0050            PUTSEC  =       $50             ;DISK PUT SECTOR DCB COMMAND
        0052            READ    =       $52             ;DISK GET SECTOR DCB COMMAND
        0057            WRITE   =       $57             ;DISK PUT SECTOR WITH READ CHECK DCB COMMAND
        0053            STATC   =       $53             ;DISK STATUS DCB COMMAND
        0021            FOMAT   =       $21             ;DISK FORMAT DCB COMMAND
        0000            NODAT   =       0               ;SIO COMMAND FOR "NO DATA" OPERATION
        0040            GETDAT  =       $40             ;SIO COMMAND FOR "DATA FROM DEVICE"
        0080            PUTDAT  =       $80             ;SIO COMMAND FOR "DATA TO DEVICE"
                        ;
                        ;       VECTORS
                        ;
                                *=      $E450
                        ;
        E450 4C EA ED           JMP     DINIT           ;DISK INIT. VECTOR
        E453 4C F0 ED           JMP     DSKIF           ;DISK INTERFACE ENTRY POINT
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;       CONSTANTS
                        ;
                                *=      $EDEA
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;***********************************************************************
                        ;       DISK INTERFACE ROUTINE STARTS HERE
                        ;***********************************************************************
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;       DISK INTERFACE INITIALIZATION ROUTINE
                        ;
```

```
EDEA A9 A0        DINIT   LDA     #160
EDEC 8D 46 02             STA     DSKTIM          SET INITIAL DISK TIMEOUT TO 160 SEC
EDEF 60                   RTS
                  ;
                  ;
                  ;
                  ;       DISK INTERFACE ENTRY POINT
                  ;
EDF0 A9 31        DSKIF   LDA     #DISKID
EDF2 8D 00 03             STA     DDEVIC          SET SERIAL BUS I.D. IN DCB
EDF5 AD 46 02             LDA     DSKTIM
EDF8 AE 02 03             LDX     DCOMND
EDFB E0 21                CPX     #FORMAT         IS COMMAND A FORMAT COMMAND?
EDFD F0 02                BEQ     PUTDTO
EDFF A9 07                LDA     #7              NO, SET TIMEOUT TO 7 SECS
EE01 8D 06 03     PUTDTO  STA     DTIMLO          PUT DISK TIMEOUT IN DCB
EE04 A2 40                LDX     #GETDAT         SET "GET DATA" COMMAND FOR SIO
EE06 A0 80                LDY     #$80            SET BYTE COUNT TO 128
EE08 AD 02 03             LDA     DCOMND          READ COMMAND IN DCB
EE0B C9 57                CMP     #WRITE          IS COMMAND A "PUT SECTOR" COMMAND?
EE0D D0 02                BNE     CKSTC
EE0F A2 80                LDX     #PUTDAT         YES, SET "PUT DATA" COMMAND FOR SIO
EE11 C9 53        CKSTC   CMP     #STATC          IS COMMAND A STATUS COMMAND?
EE13 D0 0C                BNE     PUTCNT
EE15 A9 EA                LDA     #STATVL
EE17 8D 04 03             STA     DBUFLO
EE1A A9 02                LDA     #STATVH
EE1C 8D 05 03             STA     DBUFHI          SET BUFFER ADDR TO GLOBAL STATUS BUFFER
EE1F A0 04                LDY     #4              YES, SET BYTE COUNT TO 4
EE21 8E 03 03     PUTCNT  STX     DSTATS          PUT STATUS COMMAND FOR SIO IN DCB
EE24 8C 08 03             STY     DBYTLO
EE27 A9 00                LDA     #0
EE29 8D 09 03             STA     DBYTHI          PUT BYTE COUNT IN DCB
EE2C 20 59 E4             JSR     SIOV            CALL SERIAL I/O.
EE2F 10 01                BPL     GOODST          NO ERROR
EE31 60                   RTS                     NO, GO BACK
EE32 AD 02 03     GOODST  LDA     DCOMND          READ THE COMMAND
EE35 C9 53                CMP     #STATC          WAS IT A STATUS COMMAND?
EE37 D0 0A                BNE     PUTBC
EE39 20 6D EE             JSR     PUTADR          PUT BUFFER ADDR IN TEMP REG
EE3C A0 02                LDY     #2
EE3E B1 15                LDA     (BUFADR),Y      READ DISK TIMEOUT VALUE BYTE OF STATUS
EE40 8D 46 02             STA     DSKTIM          PUT IT IN DISK TIMEOUT REG
EE43 AD 02 03     PUTBC   LDA     DCOMND
EE46 C9 21                CMP     #FORMAT         WAS COMMAND A FORMAT COMMAND?
EE48 D0 1F                BNE     ENDDIF
EE4A 20 6D EE     FMTD    JSR     PUTADR          YES, PUT BUFFER ADDR INTO TEMP REG
EE4D A0 FE                LDY     #$FE            SET BUFFER POINTER
EE4F C8           TWICE   INY
EE50 C8                   INY                     INCR BUFFER POINTER BY 2
EE51 B1 15        RDBAD   LDA     (BUFADR),Y      READ LO BYTE BAD SECTOR DATA
EE53 C9 FF                CMP     #$FF
EE55 D0 F8                BNE     TWICE           IS IT "FF" ?
EE57 C8                   INY                     YES,
EE58 B1 15                LDA     (BUFADR),Y      READ HI BYTE BAD SECTOR DATA
EE5A C8                   INY
EE5B C9 FF                CMP     #$FF
EE5D D0 F2                BNE     RDBAD           IS IT "FF" ?
EE5F 88                   DEY
EE60 88                   DEY                     YES,
```

what about $54?

```
    EE61 8C 08 03              STY     DBYTLO          PUT MAX SECTOR BYTE COUNT INTO DCB
    EE64 A9 00                 LDA     #0
    EE66 8D 09 03              STA     DBYTHI
    EE69 AC 03 03      ENDDIF  LDY     DSTATS
    EE6C 60                    RTS
                       ;
                       ;
                       ;
                       ;
                       ;       S U B R O U T I N E S
                       ;
                       ;
                       ;       PUT BUFFER ADDR FROM DCB INTO TEMP REG
                       ;
    EE6D AD 04 03      PUTADR  LDA     DBUFLO
    EE70 85 15                 STA     BUFADR
    EE72 AD 05 03              LDA     DBUFHI
    EE75 85 16                 STA     BUFADR+1        PUT BUFFER ADDR IN TEMP REG
    EE77 60                    RTS
                       ;****************************************************************
                       ;
                       ;
                       ;       SPARE BYTE OR MODULE TOO LONG FLAG
                       ;
    EE78               CNNTPC  =       *
                       ;
                               * = $14
    0014 00            DSKSPR  .BYTE   PRNORG-CNNTPC   DISKP TOO LONG
                       ;
                        .END
```

```
DISK  ***** DISKF SRC  ***** 3/9/79 ***** 4 00:00 P M


            SYMBOL  TABLE
ADDCOR  000E    ADRESS  0064    AFP     D800    ALLPOT  D208
ANTIC   D400    APPEND  0001    APPMHI  000E    ATACHR  02FB
ATAN    BE43    ATRACT  004D    AUDC1   D201    AUDC2   D203
AUDC3   D205    AUDC4   D207    AUDCTL  D208    AUDF1   D200
AUDF2   D202    AUDF3   D204    AUDF4   D206    BADTOC  0086
BADMOD  0093    BFENHI  0035    BFENLO  0034    BITMSK  006E
BLIM    028A    BLKBDV  E471    BOOT?   0009    BOOTAD  0242
BOTSCR  02BF    BPTR    003D    BRKABT  0080    BRKKEY  0011
BUFADR  0015    BUFCNT  006B    BUFRFL  0038    BUFRHI  0033
BUFRLO  0032    BUFSTR  006C    CASBUF  03FD    CASETV  E440
CASFLG  030F    CASINI  0002    CASORG  EF41    CASSBT  004B
CASSET  0040    CAUX1   023C    CAUX2   023D    CBAUDH  02EF
CBAUDL  02EE    CCOMND  023B    CDEVIC  023A    CDTMA1  0226
CDTMA2  0228    CDTMF3  022A    CDTMF4  022C    CDTMF5  022E
CDTMV1  0218    CDTMV2  021A    CDTMV3  021C    CDTMV4  021E
CDTMV5  0220    CH      02FC    CH1     02F2    CHACT   02F3
CHACTL  D401    CHAR    02FA    CHBAS   02F4    CHBASE  D409
CHKERR  003F    CHKSNT  003B    CHKSUM  0031    CHRORG  E000
CIOCHR  002F    CIOINV  E46E    CIOORG  E4A6    CIOV    E456
CIX     00F2    CKEY    004A    CKSTC   EE11    CLOSE   000C
COLAC   0072    COLBK   D01A    COLCRS  0055    COLDST  0244
COLDSV  E477    COLINC  006A    COLOR0  02C4    COLOR1  02C5
COLOR2  02C6    COLOR3  02C7    COLOR4  02C8    COLPF0  D016
COLPF1  D017    COLPF2  D018    COLPF3  D019    COLPM0  D012
COLPM1  D013    COLPM2  D014    COLPM3  D015    COLRSH  004F
CONSOL  D01F    COS     BD73    COUNTR  007E    CR      009B
CRETRY  0036    CRITIC  0042    CRNTPC  EE7B    CRSINH  02F0
CRSRIG  008D    CSOPIV  E47D    CSTAT   0288    CTIA    D000
DAUX1   030A    DAUX2   030B    DBSECT  0241    DBUFHI  0305
DBUFLO  0304    DBYTHI  0309    DBYTLO  0308    DCB     0300
DCOMND  0302    DDEVIC  0300    DEGFLG  00FB    DEPADX  0006
DELETE  0021    DELTAC  0077    DELTAR  0076    DERROR  009D
DFLAGS  0240    DIGRT   00F1    DINDEX  0057    DINIT   EDEA
DIRECT  0002    DISK    0084    DISKID  0031    DISKIV  E450
DISPLY  0003    DLISTH  D403    DLISTL  D402    DMACTL  D400
DMASK   02A0    DNACK   0041    DOSINI  000C    DOSVEC  000A
DRAWLN  0011    DRETRY  0037    DRKMSK  004E    DSKFMS  0018
DSKIF   EDF0    DSKINV  E453    DSKORG  EDEA    DSKSPD  0014
DSKTIM  0246    DSKUTL  001A    DSPFLG  02FE    DSTAT   004C
DSTATS  0303    DTIMLO  0306    DUNIT   0301    DUNUSE  0307
DVSTAT  02EA    EDITRV  E400    EEXP    00ED    ENDAIF  EE57
ENDPT   0074    EOFSRR  005B    EXPFLD  023F    ESCFLG  02A2
ESIGN   00EF    EXP     DDC0    EXP10   DDCC    FADD    DA66
FASC    D8E6    FCHRFL  00F0    FDIV    DB28    FEOF    003F
FILDAT  02FD    FILFLG  02B7    FILLIN  0012    FLD0P   DD8D
FLD0R   DD89    FLD1P   DD9C    FLD1R   DD98    FLPTR   00FC
FMOVE   DDB6    FMSFPG  0097    FMTD    FE4A    FMUL    DADB
FNCWRT  0092    FORMAT  0021    FPI     D9D2    FPREC   0006
FPSCR   05E6    FPSCR1  05EC    FPTR2   00FE    FR0     00D4
FR1     00E0    FR2     00E6    FRE     00DA    FREQ    0040
FRMADR  00CE    FRX     00EC    FSCR    05E6    FSCR1   05EC
FSTOP   DDAB    FSTOR   DDA7    FSUB    DA60    FTYPE   00ED
GETCHR  0007    GETDAT  0080    GETSEC  0005    GLBABS  02E0
GOODST  0001    GPRIOR  026F    GRACTL  D01D    GRAFM   D011
GRAFP0  D00D    GRAFP1  D00E    GRAFP2  D00F    GRAFP3  D010
HATABS  031A    HITCLR  D01E    HOLD1   0051    HOLD2   029F
HOLD3   029D    HOLD4   02BC    HOLD5   02BD    HOLDCH  007C
HPOSM0  D004    HPOSM1  D005    HPOSM2  D006    HPOSM3  D007
HPOSP0  D000    HPOSP1  D001    HPOSP2  D002    HPOSP3  D003
```

```
0108  [illegible]

HSCROL  D404   ICAX1   034A   ICAX1Z  002A   ICAX2   034B
ICAX2Z  002B   ICBAH   0345   ICBAHZ  0025   ICBAL   0344
ICBALZ  0024   ICBLH   0349   ICBLHZ  0029   ICBLL   0348
ICBLLZ  0028   ICCOM   0342   ICCOMT  0017   ICCOMZ  0022
ICDNO   0341   ICDNOZ  0021   ICHID   0340   ICHIDZ  0020
ICIDNO  002E   ICPTH   0347   ICPTHZ  0027   ICPTL   0346
ICPTLZ  0026   ICSPR   034C   ICSPRZ  002C   ICSTA   0343
ICSTAZ  0023   [illegible]    INBUFF  00F3   [illegible] 0020
INSDAT  007D   INTABS  0000   INTEMP  002D   INTINV  E46B
[illegible]    [illegible] 0010 INVFLG 02B6  IOCB    0340
IOCBAS  0020   IOCBSZ  0010   IOCFRE  00FF   IRQEN   D20E
IRQST   D20E   KBCODE  D209   KBD     0040   [illegible]
KEYBDV  E420   KEYDEL  02F1   [illegible] 00FF LBPR1 007F
LBPR2   0080   LBUFF   0580   LEDGE   0002   LINBUF  0247
LINFDS  0080   LMARGN  0052   LOCKFL  0025   LOG     DECD
LOG10   DED1   LOGCOL  0063   LOGMAP  02B2   LPENH   0234
LPENV   0235   M0PF    D000   M0PL    D008   M1PF    D001
M1PL    D009   M2PF    D002   M2PL    D00A   M3PF    D003
M3PL    D00B   MAXDEV  0021   MAXFRC  0080   MEMLO   02E7
MEMTOP  02E5   MLTTMP  0066   MODEM   0040   MONORG  F0E3
MSOMUD  001D   NEWCOL  0061   NEWROW  0060   NMIEN   D40E
NMIRES  D40F   NMIST   D40F   NOCKSM  003C   NODAT   0009
NONDEV  0082   NOTE    0026   NOTOPN  0085   NSIGN   00EE
NVALID  0084   OLDADR  005E   OLDCHR  005D   OLDCOL  005B
OLDROW  005A   OPEN    0003   OPNIN   0004   OPNINO  000C
OPNOT   0008   OPRTMP  0066   OVRRUN  008E   P0PF    D004
P0PL    D00C   P1PF    D005   P1PL    D00D   P2PF    D006
P2PL    D00E   P3PF    D007   P3PL    D00F   PACTL   D302
PADDL0  0270   PADDL1  0271   PADDL2  0272   PADDL3  0273
PADDL4  0274   PADDL5  0275   PADDL6  0276   PADDL7  0277
PBCTL   D303   PBPNT   001D   PBUFSZ  001E   PCOLR0  02C0
PCOLR1  02C1   PCOLR2  02C2   PCOLR3  02C3   PENH    D40C
PENV    D40D   PIA     D300   PLYARO  05E0   PLYEVL  DD40
PMBASE  D407   POINT   0025   POKEY   D200   POKMSK  0010
PORTA   D300   PORTB   D301   POT0    D200   POT1    D201
POT2    D202   POT3    D203   POT4    D204   POT5    D205
POT6    D206   POT7    D207   POTGO   D20B   [illegible] 0050
PRINTV  E430   PRIOR   D01B   PRNBUF  03C0   PRNORG  EE78
PRVOPN  0081   PTEMP   001F   PTIMOT  001C   PTRIG0  027C
PTRIG1  027D   PTRIG2  027E   PTRIG3  027F   PTRIG4  0280
PTRIG5  0281   PTRIG6  0282   PTRIG7  0283   PUTADR  EE6D
PUTBC   EE40   PUTCHR  000B   PUTCMT  EE21   PUTDAT  0080
PUTDTO  FF01   PUTREC  0009   PUTSEC  0050   RADFLG  00FB
[illegible] 0000 RAMLO 0004   RAMSIZ  02E4   RAMTOP  006A
RANDOM  D20A   RBLOKV  E47A   RDWAD   EE51   RDONLY  0087
READ    0052   RECVDN  0039   REDGE   0027   RENAME  0020
RMARGN  0053   ROWAC   0070   ROWCRS  0054   ROWINC  0079
RTCLOK  0012   SAVADR  0068   SAVIO   0316   SAVMSC  0058
SCREDT  0095   SCRENV  E410   SCRFLG  02BB   SCRMEM  0093
SDLSTH  0231   SDLSTL  0230   SDMCTL  022F   SENDEV  E468
SERIN   D20D   SEROUT  D20D   SETVBV  E45C   SHFAMT  006F
SHFLOK  02BE   SIN     BDB1   SIOINV  E465   SIOORG  E944
SIOV    E459   SIZEM   D00C   SIZEP0  D008   SIZEP1  D009
SIZEP2  D00A   SIZEP3  D00B   SKCTL   D20F   SKRES   D20A
SKSTAT  D20F   SOUNDR  0041   [illegible] 0008 [illegible] BEB1
SRTIMR  022B   SSFLAG  02FF   SSKCTL  0232   STACKP  0318
STATE   0053   STATIS  000D   STATUS  0030   STATVR  0002
STATVL  00EA   STICK0  0278   STICK1  0279   STICK2  027A
STICK3  027B   STIMER  D209   STRIG0  0284   STRIG1  0285
STRIG2  0286   STRIG3  0287   [illegible] 029E SUCCES 0001
```

```
[illegible] 000A     [illegible] 0000     SKCTL   D20F     SKRES   D20A
SKSTAT  D20F     SOUNDR  0041     [illegible] 00DE     [illegible] 0081
SRTIMR  022B     SSFLAG  02FF     SSKCTL  0232     STACKP  0318
STATE   0053     STAT1B  000D     STATUS  0030     STATVH  0002
STATVL  00EA     STICK0  0278     STICK1  0279     STICK2  027A
STICK3  027B     STIMER  D209     STRIG0  0284     STRIG1  0285
STRIG2  0286     STRIG3  0287     [illegible] 029E     [illegible] 0001


DISK   ***** DISKF SRC   ***** [illegible]/79 ***** 4:00:00 P M

SWPFLG  007B     SYSVBV  E45F     TABMAP  02A3     TEMP    020E
TEMP1   0312     TEMP2   0314     TEMP3   0315     TIMER1  030C
TIMER2  0310     TIMFLG  0317     TIMOUT  008A     TINDEX  0293
TMPCHR  0050     TMPCOL  02B9     TMPLBT  02A1     TMPROW  02B8
TMPX1   029C     TRAMSZ  0006     TRIG0   D010     TRIG1   D011
TRIG2   D012     TRIG3   D013     TRNWCD  0029     TSTAT   0319
TSTDAT  0007     TWICE   E8F?     TXTCOL  0291     TXTMSC  0294
TXTOLD  0296     TXTROW  0290     [illegible] 0024     USAREA  0480
VBREAK  0206     VCOUNT  D40B     VCTABL  E480     VDELAY  D01C
VDSLST  0200     VECTBL  E400     VIMIRQ  0216     VINTER  0204
VKEYBD  0208     VPRCED  0202     VSCROL  D405     VSERIN  020A
VSEROC  020E     VSEROR  020C     VTIMR1  0210     VTIMR2  0212
VTIMR4  0214     VVBLKD  0224     VVBLKI  0222     WARMST  0008
WARMSV  E474     WMODE   02B9     WRITE   0057     WRONLY  0008
WSYNC   D40A     XITVBV  E462     XMTDON  003A     ZIOCB   0020
ZTEMP1  00F5     ZTEMP3  00F9     ZTEMP4  00F7
```

```
                           ;
                           .TITLE 'PRINTER  PH.SRC  PRINTP.SRC  ***** 3/7/79 ***** 4.00 00 E R '
                           ;
                           ;
                           ;
                           ;
                           ;
                           ;
                           ;
                           ;
                           ;       DEVICE NUMBER OR CODE EQUATES
                           ;
      0008                 OPNOUT  =       $8              IOCB OPEN FOR OUTPUT COMMAND
      0028                 NBUFSZ  =       40              PRINT NORMAL BUFFER SIZE
      0014                 DBUFSZ  =       20              PRINT DOUBLE BUFFER SIZE
      001D                 SBUFSZ  =       29              PRINT SIDEWAYS BUFFER SIZE
      0040                 PDEVN   =       $40             PRINTER DEVICE NUMBER
      0053                 STATC   =       $53             DCB STATUS COMMAND CODE
      0057                 WRITEC  =       $57             DCB WRITE COMMAND
      009B                 CR      =       $9B             ASCII CARRIAGE RETURN
      0020                 SPACE   =       $20             ASCII SPACE CHAR.
      004E                 N       =       $4E             ASCII "N" CHAR
      0044                 D       =       $44             ASCII "D" CHAR
      0053                 S       =       $53             ASCII "S" CHAR
                           ;
                           ;
                           ;       PRINTER HANDLER ENTRY POINTS
                           ;
                           ;
                           ;
                           ;
                                   *= $E430
                           ;
     E430 9E EE                    .WORD   PHOPEN-1        PRINTER HANDLER OPEN
     E432 DB EE                    .WORD   PHCLOS-1        PH CLOSE
     E434 9D EE                    .WORD   BADST-1 PH READ
     E436 A6 EE                    .WORD   PHWRIT-1        PH WRITE
     E438 80 EE                    .WORD   PHSTAT-1        PH STATUS
     E43A 9D EE                    .WORD   BADST-1 PH SPECIAL
     E43C 4C 78 EE                 JMP     PHINIT          PH INIT.
     E43F 00                       .BYTE   0               ROM FILLER
                           ;
                           ;
                           ;
                           ;
                           ;
                                   *= PRNORG
                           ;
                           ;
                           ;
                           ;
                           ;       PRINTER HANDLER INITIALIZATION ROUTINE
                           ;
   EE78 A9 1E              PHINIT  LDA     #30
   EE7A 85 1C                      STA     PTIMOT          SET UP INITIAL PRINTER TIMEOUT OF 30 SEC
   EE7C 60                         RTS
                           ;
```

```
                        ;
                        ;         PRINTER HANDLER CONSTANTS
                        ;
        EE7D EA 02      PHSTLO  .WORD   DVSTAT          STATUS BUFFER POINTER
        EE7F C0 03      PHCHLO  .WORD   PRNBUF          CHAR. BUFFER POINTER
                        ;
                        ;
                        ;
                        ;         ************************
                        ;         PRINTER HANDLER ROUTINES
                        ;         ************************
                        ;
                        ;
                        ;
                        ;
                        ;
                        ;         PRINTER HANDLER STATUS ROUTINE
                        ;
        EE81 A9 04      PHSTAT  LDA     #4
        EE83 85 1E              STA     PBUFSZ          SET BUFFER SIZE TO 4 BYTES
        EE85 AE 7D EE           LDX     PHSTLO
        EE88 AC 7E EE           LDY     PHSTLO+1        SET POINTER TO STATUS BUFFER
        EE8B A9 53              LDA     #STATC          SET COMMAND TO "STATUS"
        EE8D 8D 02 03           STA     DCOMND          SET STATUS COMMAND
        EE90 8D 0A 03           STA     DAUX1
        EE93 20 E6 EE           JSR     SETDCB          GO SETUP DCB
        EE96 20 59 E4           JSR     SIOV            SEND STATUS COMMAND
        EE99 30 03              BMI     BADST           GO IF ERROR
        EE9B 20 14 EF           JSR     PHPUT           YES,PUT STATUS INTO GLOBAL BUFFER.
        EE9E 60         BADST   RTS
                        ;
                        ;
                        ;
                        ;
                        ;         PRINTER HANDLER OPEN ROUTINE
                        ;
        EE9F 20 81 EE   PHOPEN  JSR     PHSTAT          DO STATUS COMMAND TO SIO
        EEA2 A9 00              LDA     #0
        EEA4 85 1D              STA     PBPNT           CLEAR PRINT BUFFER POINTER
        EEA6 60                 RTS
                        ;
                        ;
                        ;
                        ;         PRINTER HANDLER WRITE ROUTINE
                        ;
        EEA7 85 1F      PHWRIT  STA     PTEMP           SAVE ACCUM
        EEA9 20 1A EF           JSR     PRMODE          GO DETERMINE PRINT MODE
        EEAC A6 1D              LDX     PBPNT
        EEAE A5 1F              LDA     PTEMP           GET CHAR. SENT BY CIO
        EEB0 9D C0 03           STA     PRNBUF,X        PUT CHAR. IN PRINT BUFFER
        EEB3 E8                 INX                     INCR. BUFFER POINTER
        EEB4 E4 1E              CPX     PBUFSZ          BUFFER POINTER=BUFFER SIZE?
        EEB6 F0 13              BEQ     BUFFUL
        EEB8 86 1D              STX     PBPNT           SAVE BUFFER POINTER
        EEBA C9 9B              CMP     #CR             IS CHAR. = EOL ?
        EEBC F0 03              BEQ     BLFILL          IF YES, GO DO BLANK FILL.
        EEBE A0 01              LDY     #SUCCES         PUT GOOD STATUS IN Y REG FOR CIO.
        EEC0 60                 RTS
        EEC1 A9 20      BLFILL  LDA     #SPACE          PUT BLANK IN ACCUM.
```

problem if mode changed in mid line!

```
  EEC3 99 C0 03     FILLBF  STA     PRNBUF,Y          STORE IT IN PRINT BUFFER
  EEC6 C8                   INY
  EEC7 E4 1E                CPX     PBUFSZ
  EEC9 D0 F8                BNE     FILLBF          BUFFER BLANK FILLED?
  EECB A9 00        BUFFIL  LDA     #0
  EECD 85 1D                STA     PBPNT           CLEAR PRINT BUFFER POINTER
  EECF AE 7C EE             LDX     PRCHLO
  EED2 AC 00 EE             LDY     PRCHLO+1        SET POINTER TO PRINT BUFFER
  EED5 20 E6 EE             JSR     SETDCB          GO SETUP DCB
  EED8 20 59 E4             JSR     SIOV            SEND PRINT COMMAND
  EEDB 60                   RTS                     YES.
                    ;
                    ;
                    ;
                    ;
                    ;       PRINTER HANDLER CLOSE ROUTINE
                    ;
  EEDC 20 1A EF     PRCLOS  JSR     PRMODE          GO DETERMINE PRINT MODE
  EEDF A6 1D                LDX     PBPNT
  EEE1 D0 DE                BNE     BLFILL
  EEE3 A0 01                LDY     #SUCCES
  EEE5 60                   RTS
                    ;
                    ;
                    ;
                    ;
                    ;
                    ;
                    ;       S U B R O U T I N E S
                    ;
                    ;
                    ;
                    ;
                    ;
                    ;       SET UP DCB TO CALL SIO
                    ;
  EEE6 8E 04 03     SETDCB  STX     DBUFLO
  EEE9 8C 05 03             STY     DBUFHI          SET BUFFER POINTER
  EEEC A9 40                LDA     #PDEVN
  EEEE 8D 00 03             STA     DDEVIC          SET PRINTER BUS I.D. FOR DCB
  EEF1 A9 01                LDA     #1
  EEF3 8D 01 03             STA     DUNIT           SET UNIT NUMBER TO 1
  EEF6 A9 80                LDA     #$80            DEVICE WILL EXPECT DATA
  EEF8 AE 02 03             LDX     DCOMND
  EEFB E0 53                CPX     #STATC          STATUS COMMAND?
  EEFD D0 02                BNE     PSIOC
  EEFF A9 40                LDA     #$40            EXPECT DATA FROM DEVICE
  EF01 8D 03 03     PSIOC   STA     DSTATS          SET SIO MODE COMMAND
  EF04 A5 1E                LDA     PBUFSZ
  EF06 8D 08 03             STA     DBYTLO          SET LO BYTE COUNT
  EF09 A9 00                LDA     #0
  EF0B 8D 09 03             STA     DBYTHI          SET HI BYTE COUNT
  EF0E A6 1C                LDX     PTIMOT
  EF10 8E 06 03             STX     DTIMLO          SET DEVICE TIMEOUT COUNT
  EF13 60                   RTS
                    ;
                    ;
                    ;
```

```
                        ;
                        ;      GET DEVICE TIMEOUT FROM STATUS & SAVE IT
                        ;
EF14 AD EC 02           PTMOUT  LDA     DVSTAT+2
EF17 85 1C                      STA     PTIMOT          SAVE DEVICE TIMEOUT
EF19 60                         RTS
                        ;
                        ;
                        ;
                        ;
                        ;      DETERMINE PRINT MODE & SETUP PRINT BUFFER SIZE, DCB PRINT
                        ;              COMMAND, & DCB AUX1 FOR PRINT MODE
                        ;
EF1A A0 57              PRMODE  LDY     #WRITEC         PUT WRITE COMMAND IN Y REG
EF1C A5 2B                      LDA     ICAX2Z          READ PRINT MODE
EF1E C9 4E              CMODE   CMP     #N
EF20 D0 04                      BNE     CDUBL           PRINT NORMAL ?
EF22 A2 28                      LDX     #NBUFSZ         YES, SET NORMAL CHAR. BUFFER SIZE
EF24 D0 0E                      BNE     SETBSZ
EF26 C9 44              CDUBL   CMP     #D
EF28 D0 04                      BNE     CSIDE           PRINT DOUBLE?
EF2A A2 14                      LDX     #DBUFSZ         YES, SET DOUBLE CHAR. BUFFER SIZE
EF2C D0 06                      BNE     SETBSZ
EF2E C9 53              CSIDE   CMP     #S              PRINT SIDEWAYS ?
EF30 D0 0B                      BNE     GOERR           IF NOT, GO TO ERROR ROUTINE
EF32 A2 1D                      LDX     #SBUFSZ         YES, SET SIDEWAYS BUFFER SIZE
EF34 86 1E              SETBSZ  STX     PBUFSZ          STORE PRINT BUFFER SIZE
EF36 8C 02 03                   STY     DCOMND          STORE DCB COMMAND
EF39 8D 0A 03                   STA     DAUX1           STORE DCB AUX1 PRINT MODE
EF3C 60                         RTS
EF3D A9 4E              GOERR   LDA     #N              SET DEFAULT PRINT MODE TO NORMAL
EF3F D0 DD                      BNE     CMODE
                        ;*************************************************************
                        ;
                        ;
                        ;      SPARE BYTE OR MODULE TOO LONG FLAG
                        ;
EF41                    CRNTPR  =       *
                        ;
                                *=  $14
                        ;
0014 DE                 PRNSPR  .BYTE   CASORG-CRNTPR   PRINTR TOO LONG
                        ;
                        .END
```

```
PRIMIPI  *****  TRINTP.SRC  *****  9/3/79  *****  4:00:00 P.M.

          SYMBOL  TABLE
ABUEDM   [illegible]    ADRESS   0064     ALE      D000     ALLPOT   D208
ANTIC    D400     APPEND   0001     APPMHI   000E     ATACHR   02FB
ATAN     DE43     ATRACT   004D     AUDC1    D201     AUDC2    D203
AUDC3    D205     AUDC4    D207     AUDCTL   D208     AUDF1    D200
AUDF2    D202     AUDF3    D204     AUDF4    D206     BADTOC   0096
BADMOD   0091     BADBT    FF9F     BFENHI   0035     BFENLO   0034
BITMSK   006E     BLFILL   EEC1     BLIM     028A     BLKBDV   E471
BOOT?    0009     BOOTAD   0242     BOTSCR   02BF     BPTR     003D
BRKABT   0080     BRKKEY   0011     BUFADR   0016     BUFCNT   006B
BUFFUL   EEC0     BUFRFL   0038     BUFRHI   0033     BUFRLO   0032
BUFSTR   006C     CASBUF   03FD     CASETV   E440     CASFLG   030F
CASINI   0002     CASORG   EF41     CASSBT   004B     CASSET   0043
CAUX1    023C     CAUX2    023D     CBAUDH   02EF     CBAUDL   02EE
CCOMND   023B     CDEVIC   023A     CDTMA1   0226     CDTMA2   0228
CDTMF3   022A     CDTMF4   022C     CDTMF5   022E     CDTMV1   0218
CDTMV2   021A     CDTMV3   021C     CDTMV4   021E     CDTMV5   0220
CDUBL    EFA9     CH       02FC     CH1      02F2     CHACT    02F3
CHACTL   D401     CHAR     02FA     CHBAS    02F4     CHBASE   D409
CHKERR   000F     CHKSNT   003B     CHKSUM   0031     CHRORG   E000
CIOCHR   002F     CIOINV   E46E     CIOORG   E4A6     CIOV     E456
CIX      00F2     CKEY     004A     CLOSE    000C     CMDONE   EF1E
COLAC    0072     COLBK    D01A     COLCRS   0055     COLDST   0244
COLDSV   E477     COLINC   007A     COLOR0   02C4     COLOR1   02C5
COLOR2   02C6     COLOR3   02C7     COLOR4   02C8     COLPF0   D016
COLPF1   D017     COLPF2   D018     COLPF3   D019     COLPM0   D012
COLPM1   D013     COLPM2   D014     COLPM3   D015     COLRSH   004F
CONSOL   D01F     COS      BD73     COUNTR   007E     CR       009B
CRETRY   0036     CRITIC   0042     CRNTPC   EE41     CRSINH   02F0
CRSRDR   000D     CSIDE    EF2E     CSOPIV   E470     CSTAT    0288
CTIA     D000     D        0044     DAUX1    030A     DAUX2    030B
DBSECT   0241     DBUFHI   0305     DBUFLO   0304     DBUFSZ   0014
DBYTHI   0309     DBYTLO   0308     DCB      0300     DCOMND   0302
DDEVIC   0300     DEGFLG   00FB     DEGON    0006     DELETE   0021
DELTAC   0077     DELTAR   0076     DERROR   0090     DFLAGS   0240
DIGRT    00F1     DINDEX   0057     DIRECT   0002     DISK     0044
DISKIV   E450     DISPLY   0050     DLISTH   D403     DLISTL   D402
DMACTL   D400     DMASK    02A0     DNACK    008B     DOSINI   000C
DOSVEC   000A     DRAWLN   0011     DRETRY   0037     DRKMSK   004E
DSKFMS   0018     DSKINV   E453     DSKORG   EDEA     DSKTIM   0246
DSKUTL   001A     DSPFLG   02FE     DSTAT    004C     DSTATS   0303
DTIMLO   0306     DUNIT    0301     DUNUSE   0307     DVSTAT   02EA
EDITRV   E400     EEXP     00ED     ENDPT    0074     EOFFHR   0098
ERRFLG   023E     ESCFLG   02A2     ESIGN    00EF     EXP      DDC0
EXP10    DDCC     FADD     DA66     FASC     D8E6     FCHRFL   00F0
FDIV     DB28     FEOF     003F     FILDAT   02FD     FILFLG   02B7
FILLDF   EEC3     FILLIN   0012     FLD0P    DD8D     FLD0R    DD89
FLD1P    DD9C     FLD1R    DD98     FLPTR    00FC     FMOVE    DDB6
FMSZPO   0043     FMUL     DADB     FNENDT   0092     FORMAT   0022
FPI      D8B2     FPREC    0006     FPSCR    05E6     FPSCR1   05EC
FPTR2    00FE     FR0      00D4     FR1      00E0     FR2      00E6
FRE      00DA     FREQ     0040     FRMERR   008C     FRX      00EC
FSCR     05E6     FSCR1    05EC     FST0P    DDAB     FST0R    DDA7
FTWO     DA60     FTYPE    00CA     GETCHR   000T     GETNTC   D005
GLBABS   02E0     GOERR    FF30     GPRIOR   026F     GRACTL   D01D
GRAFM    D011     GRAFP0   D00D     GRAFP1   D00E     GRAFP2   D00F
GRAFP3   D010     HATABS   031A     HUYCLR   001E     HOLD1    0051
HOLD2    029F     HOLD3    029H     HOLD4    02BC     HOLD5    0289
HOLDCH   007C     HPOSM0   D004     HPOSM1   D005     HPOSM2   D006
HPOSM3   D007     HPOSP0   D000     HPOSP1   D001     HPOSP2   D002
```

```
PRINTER  *****  PRINTP SRC  ***** 3/9/79 ***** 4:00:00 P.M.

HPOSP3   D003     HSCROL  D404     ICAX1    034A     ICAX1Z   002A
ICAX2    034B     ICAX2Z  002B     ICBAH    0345     ICBAHZ   0025
ICBAL    0344     ICBALZ  0024     ICBLH    0349     ICBLHZ   0029
ICBLL    0348     ICBLLZ  0028     ICCOM    0342     ICCOMT   0017
ICCOMZ   0022     ICDNO   0341     ICDNOZ   0021     ICHID    0340
ICHIDZ   0020     ICIDNO  002E     ICPTH    0347     ICPTHZ   0027
ICPTL    0346     ICPTLZ  0026     ICSPR    034C     ICSPRZ   002C
ICSTA    0343     ICSTAZ  0023     IFP      D9AA     INBUFF   00F3
INSCLR   0020     INSDAT  007D     INIABS   0200     INTEMP   022D
INTINV   E960     INTORG  E6D5     IN77BS   0010     INVFLG   02B6
IOCB     0340     IOCBAS  0020     IOCBSZ   0010     IOCFRE   00FF
IRQEN    D20E     IRQST   D20E     KBCODE   D209     KBD      0048
KBDVEC   F3F4     KEYBDV  E420     KEYDEL   02F1     LBFEND   05FF
LBPR1    057E     LBPR2   057F     LBUFF    0580     LEDGE    0002
LINBUF   0247     LIN2BG  0000     LMARGN   0052     LOCKFL   0023
LOG      DECD     LOG10   DED1     LOGCOL   0063     LOGMAP   02B2
LPENH    0234     LPENV   0235     M0PF     D000     M0PL     D008
M1PF     D001     M1PL    D009     M2PF     D002     M2PL     D00A
M3PF     D003     M3PL    D00B     MAXDEV   0021     MAXIOC   0080
MEMLO    02E7     MEMTOP  02E5     MLTTMP   0066     MODEM    0040
MONORG   F0F3     MYDMOD  0010     N        004E     NBUFSZ   002B
NEWCOL   0061     NEWROW  0060     NMIEN    D40E     NMIRES   D40F
NMIST    D40F     NOCKSM  003C     NUMDEV   0082     NOTE     0026
NOTOPN   0085     NSIGN   00EE     NVALID   0084     OLDADR   005E
OLDCHR   005D     OLDCOL  005B     OLDROW   005A     OPEN     0003
OPNIN    0004     OPNINO  000C     OPNOT    0006     OPNOUT   0002
OPNTMP   0066     OVRRUN  008E     P0PF     D004     P0PL     D00C
P1PF     D005     P1PL    D00D     P2PF     D006     P2PL     D00E
P3PF     D007     P3PL    D00F     PACTL    D302     PADDL0   0270
PADDL1   0271     PADDL2  0272     PADDL3   0273     PADDL4   0274
PADDL5   0275     PADDL6  0276     PADDL7   0277     PBCTL    D303
PBPNT    001D     PBUFSZ  001E     PCOLR0   02C0     PCOLR1   02C1
PCOLR2   02C2     PCOLR3  02C3     PDEVN    0040     PENH     D40C
PENV     D40D     PHCHLO  EE7F     PHCLOS   EEDC     PHINIT   EE78
PHOPEN   EE3E     PHPUT   EF14     PHSTAT   EEB1     PHSTLO   EE7D
PHWRIT   EEA7     PIA     D300     PLYARD   05E0     PLYEVL   D040
PMBASE   D407     POINT   0055     POKEY    D200     POKMSK   0010
PORTA    D300     PORTB   D301     POT0     D200     POT1     D201
POT2     D202     POT3    D203     POT4     D204     POT5     D205
POT6     D206     POT7    D207     POTGO    D20B     PRINTB   0030
PRINTV   E430     PRIOR   D01B     PRNODE   EF1A     PRNBUF   03C0
PRNORG   EE78     PRNSPR  0014     PRVUPN   0081     PSIOC    EED1
PTEMP    001F     PTIMOT  0014     PTRIG0   027C     PTRIG1   027D
PTRIG2   027E     PTRIG3  027F     PTRIG4   0280     PTRIG5   0281
PTRIG6   0282     PTRIG7  0283     PUTCHR   0008     PUTREC   0007
RADFLG   00FB     RADON   0000     RAMLO    0004     RAMSIZ   02E4
RAMTOP   006A     RANDOM  D20A     RBLOKV   E47A     RDONLY   0087
RECVDN   0039     REDGE   0027     RENAME   0020     RMARGN   0053
ROWAC    0070     ROWCRS  0054     RUNINC   0079     RTCLOK   0012
S        0003     SAVADR  0068     SAVIO    0316     SAVMSC   0058
SBUFSZ   001D     SCREDT  0045     SCRENV   E410     SCRFLG   02BB
SCRMEM   0058     SDLSTH  0231     SDLSTL   0230     SDMCTL   022F
SENDEV   E468     SERIN   D20D     SEROUT   D20D     SETMBZ   EF39
SETVCB   FFDA     SETVBV  E45C     SHFAMT   006F     SHFLOK   02BE
SIN      0001     SIOINV  E46B     SIOVRB   E944     SIOV     E459
SIZEM    D00C     SIZEP0  D008     SIZEP1   D009     SIZEP2   D00A
SIZEP3   D00B     SKCTL   D20F     SKREST   D20A     SKSTAT   D20F
SOUNDR   0041     SPACE   0020     SPMTIL   000E     SQR      BEB1
SRTIMR   022B     SSFLAG  02FF     SSKCTL   0232     STACKP   0318
STATE    0002     STATTS  000D     STATUS   0030     STICK0   0278
```

```
PRINTED ***** PRINTP.SRC ***** 3/9/79 ***** 4:00:00 P.M

STICK1  0279      STICK2  027A      STICK3  027B      STIMER  0209
STRIG0  0284      STRIG1  0285      STRIG2  0286      STRIG3  0287
SUBTMP  029E      SUCCES  0001      SWPFLG  007B      SYSVBV  E45F
TABMAP  02A3      TEMP    020F      TEMP1   0312      TEMP2   0314
TEMP3   0315      TIMER1  030C      TIMER2  0310      TIMFLG  0317
TIMOUT  00BA      TINDEX  0293      TMPCHR  0050      TMPCOL  0289
TMPLBT  02A1      TMPROW  02B8      TMPX1   029C      TRAMSZ  0006
TRIG0   D010      TRIG1   D011      TRIG2   D012      TRIG3   D013
TRNRCD  0089      TSTAT   0319      TSTDAT  0007      TXTCOL  0291
TXTMSC  0294      TXTOLD  0296      TXTROW  0290      UNLOCK  0024
USAREA  0480      VBREAK  0206      VCOUNT  D40B      VCTABL  E480
VDELAY  D01C      VDSLST  0200      VECTBL  E400      VIMIRQ  0216
VINTER  0204      VKEYBD  0208      VPRCED  0202      VSCROL  D405
VSERIN  020A      VSEROC  020E      VSEROR  020C      VTIMR1  0210
VTIMR2  0212      VTIMR4  0214      VVBLKD  0224      VVBLKI  0222
WARMST  0008      WARMSV  E474      WMODE   0289      WRITEC  0057
WRONLY  0083      WSYNC   D40A      XITVBV  E462      XMTDON  003A
ZIOCB   0020      ZTEMP1  00F5      ZTEMP3  00F9      ZTEMP4  00F7
```

```
                                  TITLE  'CASSET HANDLER 3/12 (DK1:CASCV)'
0003               CBUFH    =            CASBUF/256
00FD               CBUFL    =            -256*CBUFH+CASBUF
0040               SRSTA    =            $40       SIO READ STATUS
0080               SWSTA    =            $80       SIO WRITE STATUS
0034               MOTRGO   =            $34
003C               MOTRST   =            $3C
                   ;
                   ;
00FC               DTA      =            $FC      DATA RECORD TYPE BYTE
00FA               DT1      =            $FA      LAST DATA RECORD
00FE               EOT      =            $FE      END OF TAPE      ?
00FB               HDR      =            $FB      HEADER
0002               TONE1    =            2         CHANGE TO RECORD MODE TONE
0001               TONE2    =            1         PRESS PLAY TONE
                   ;
                   ;
                   ;
                            *=CASETV
E440 4B EF 3A               .WORD   OPENC-1,CLOSEC-1,GBYTE-1,PBYTE-1,STATU-1,SPECIAL-1
E443 F0 D5 EF
E446 0F F0 27
E449 F0 4A EF
E44C 4C 41 EF               JMP     INIT
E44F 00                     .BYTE   0                 ROM FILLER BYTE
                   ;
                   ;
                   ;
                   ; USED IN MONITP FOR CASSETTE BOOT
                   ;
                            *=RBLOKV
E47A 4C E9 EF               JMP     RBLOK
                   ;
                            *=CSOPIV
E47D 4C 5D EF               JMP     OPINP
                   ;
                   ;
                            *=CASORG
                   ;
                   ;
                   ; INIT ROUTINE
                   ;
EF41 A9 CC         INIT     LDA     #$CC
EF43 8D EE 02               STA     CBAUDL
EF46 A9 05                  LDA     #$05
EF48 8D EF 02               STA     CBAUDH   SET CASSET BAUD RATE TO 600
EF4B 60            SPECIAL  RTS              THATS ALL FOLKS
```

```
                        ;
                        ; OPEN FUNCTION - WITH NO TIMING ADJUST
                        ;
EF4C A5 2B              OPENC   LDA     ICAX2Z  GET AX2
EF4E 85 3E                      STA     FTYPE   SAVE IT FOR FUTURE REFERENCE
EF50 A5 2A                      LDA     ICAX1Z
EF52 29 0C                      AND     #$0C    IN AND OUT BITS
EF54 C9 04                      CMP     #$04
EF56 F0 05                      BEQ     OPINP
EF58 C9 08                      CMP     #$08    SEE IF OPEN FOR OUTPUT
EF5A F0 39                      BEQ     OPOUT
EF5C 60                         RTS             IF ALREADY OPEN, RETURN LEAVING STATUS=$84 — $92
EF5D A9 00              OPINP   LDA     #0
EF5F 8D 89 02                   STA     WMODE   SET READ MODE
EF62 85 3F                      STA     FEOF    NO EOF YET
EF64 A9 01              SFH     LDA     #TONE2  TONE FOR PRESS PLAY
EF66 20 58 FD                   JSR     BEEP    GO BEEP
EF69 30 24                      BMI     OPNRTN          IF ERROR DURING BEEP
EF6B A9 34                      LDA     #MOTRGO
EF6D 8D 02 D3                   STA     PACTL   TURN MOTOR ON
EF70 A0 40                      LDY     #$40            5-31-79   9 SEC READ LEADER
EF72 A2 02                      LDX     #2
EF74 A9 03                      LDA     #3
EF76 8D 2A 02                   STA     CDTMF3
EF79 20 5C E4                   JSR     SETVBV          SET UP VBLANK TIMER
EF7C AD 2A 02           WAITTM  LDA     CDTMF3
EF7F D0 FB                      BNE     WAITTM          WAIT FOR MOTOR TO COME UP TO SPEED
EF81 A9 80                      LDA     #$80    NEXT  BYTE=NO BYTES IN BUFFER
EF83 85 3D                      STA     BPTR
EF85 8D 8A 02                   STA     BLIM
EF88 4C D3 EF                   JMP     OPOK    OPEN OK
                        ;
                        ; OPEN FOR OUTPUT
                        ;
EF8B A0 80              PORM    LDY     #BRKABT BREAK KEY ABORT STATUS
EF8D C6 11                      DEC     BRKKEY  RESET BREAK KEY
EF8F A9 00              OPNRTN  LDA     #0      CLEAR WRITE MODE FLAG
EF91 8D 89 02                   STA     WMODE
EF94 60                         RTS             AND EXIT.
                        ;
EF95 A9 80              OPOUT   LDA     #$80
EF97 8D 89 02                   STA     WMODE   SET WRITE MODE
EF9A A9 02                      LDA     #TONE1  TELL USER TO TURN ON RECORD MODE
EF9C 20 58 FD                   JSR     BEEP
EF9F 30 EE                      BMI     OPNRTN          IF ERROR DURING BEEP
EFA1 A9 CC                      LDA     #$CC    SET BAUD RATE
EFA3 8D 04 D2                   STA     AUDF3   WHICH SEEMS TO BE NESSECARY
EFA6 A9 05                      LDA     #$05    FOR SOME OBSCURE REASON
EFA8 8D 06 D2                   STA     AUDF4
EFAB A9 60                      LDA     #$60
EFAD 8D 00 03                   STA     DDEVIC
EFB0 20 68 E4                   JSR     SENDEV  TELL POKEY TO WRITE MARKS
EFB3 A9 34                      LDA     #MOTRGO WRITE X SEC BLANK TAPE
EFB5 8D 02 D3                   STA     PACTL
EFB8 A9 03                      LDA     #3
EFBA A2 04                      LDX     #4      5/30/79 20 SEC LEADER
EFBC A0 80                      LDY     #$80
EFBE 20 5C E4                   JSR     SETVBV
EFC1 A9 FF                      LDA     #$FF
EFC3 8D 2A 02                   STA     CDTMF3
```

```
        EFC6 A5 11          WDLR    LDA     BRKKEY
        EFC8 F0 C1                  BEQ     PBRK     IF BREAK DURING WRITE LEADER
        EFCA AD 2A 02               LDA     CDTMF3
        EFCD D0 F7                  BNE     WDLR
        EFCF A9 00                  LDA     #0       INIT BUFFER POINTER
        EFD1 85 3D                  STA     BPTR
        EFD3 A0 01          GPOK    LDY     #SUCCES
        EFD5 60                     RTS
```

```
                        ;
                        ; GET BYTE
                        ;
EFD6 A5 3F              GBYTE   LDA     FEOF    IF AT EOF ALREADY
EFD8 30 33                      BMI     ISEOF   RETURN EOF STATUS
EFDA A6 3D                      LDX     BPTR    BUFFER POINTER
EFDC EC 8A 02                   CPX     BLIM    IF END OF BUFFER
EFDF F0 08                      BEQ     RBLOK   READ ANOTHER BLOCK
EFE1 BD 00 04                   LDA     CASBUF+3,X  GET NEXT BYTE
EFE4 E6 3D                      INC     BPTR    BUMP POINTER
EFE6 A0 01                      LDY     #SUCCES OK STATUS
EFE8 60                 GBX     RTS
EFE9 A9 52              RBLOK   LDA     #'R     READ OPCODE
EFEB 20 95 FD                   JSR     SIOSB   SIO ON SYS BUF
EFEE 98                         TYA
EFEF 30 F7                      BMI     GBX     IF SIO ERRORS, RETURN
EFF1 A9 00                      LDA     #0
EFF3 85 3D                      STA     BPTR    RESET POINTER
EFF5 A2 80                      LDX     #$80    DEFAULT # BYTES
EFF7 AD FF 03                   LDA     CASBUF+2
EFFA C9 FE                      CMP     #EOT
EFFC F0 0D                      BEQ     ATEOF   IF HEADER, GO READ AGAIN
EFFE C9 FA                      CMP     #DT1    IF LAST DATA REC
F000 D0 03                      BNE     NLR
F002 AE 7F 04                   LDX     CASBUF+130      LAST DATA RECORD, GET # BYTES
F005 8E 8A 02           NLR     STX     BLIM
F008 4C D6 EF                   JMP     GBYTE   GET NEXT BYTE
F00B C6 3F              ATEOF   DEC     FEOF    SET FEOF
F00D A0 88              ISEOF   LDY     #EOFERR ENDFILE STATUS
F00F 60                         RTS
```

```
                        ;
                        ; PUT BYTE TO BUFFER
                        ;
   F010 A6 3D           PBYTE   LDX     BPTR     BUFFER POINTER
   F012 9D 00 04                STA     CASBUF+3,X       STORE CHAR AWAY
   F015 E6 3D                   INC     BPTR     BUMP POINTER
   F017 A0 01                   LDY     #SUCCES OK STATUS
   F019 E0 7F                   CPX     #127     IF BUFFER FULL
   F01B F0 01                   BEQ     *+3
   F01D 60                      RTS
                        ; WRITE OUT THE BUFFER
   F01E A9 FC                   LDA     #DTA     RECORD TYPE = DATA
   F020 20 02 F0                JSR     WSIOSB   DO WRITE ON SYSTEM BUFFER
   F023 A9 00                   LDA     #0
   F025 85 3D                   STA     BPTR     RESET BUFFER POINTER
   F027 60                      RTS              EXIT
```

```
                        ;
                        ; STATUS - RETURN STATUS INFO THRU DVSTAT  ?
                        ;
    F028 A0 01          STATU   LDY     #SUCCES
    F02A 60                     RTS
```

```
                        ;
                        ; CLOSE
                        ;
F02B AD 89 02           CLOSEC  LDA     WMODE   BR IF WRITING
F02E 30 08                      BMI     CLWRT   GO CLOSE FOR WRITE
                        ;CLOSE FOR READ - FLAG CLOSED
F030 A0 01                      LDY     #SUCCES         SUCCESSFULL
F032 A9 3C              FCAX    LDA     #MOTRST STOP THE MOTOR IN CASE WAS SHORT IRG MODE
F034 8D 02 03                   STA     PACTL
F037 60                         RTS
F038 A6 3D              CLWRT   LDX     BPTR    BUFFER POINTER
F03A F0 0A                      BEQ     WTLR            IF NO DATA BYTES IN BUFFER, NO DT1 REC
F03C 8E 7F 04                   STX     CASBUF+130      WRITE TO LAST RECORD
F03F A9 FA                      LDA     #DT1    REC TYPE
F041 20 D2 F0                   JSR     WSIOSB  WRITE OUT USER BUFFER
F044 30 EC                      BMI     FCAX    GO IF ERROR
F046 A2 7F              WTLR    LDX     #127    ZERO BUFFER
F048 A9 00                      LDA     #0
F04A 9D 00 04           ZTBUF   STA     CASBUF+3,X
F04D CA                         DEX
F04E 10 FA                      BPL     ZTBUF
F050 A9 FE                      LDA     #EOT    WRITE EOT RECORD
F052 20 D2 F0                   JSR     WSIOSB
F055 4C 32 F0                   JMP     FCAX    FLAG CLOSED AND EXIT)
```

— is this — PILOT stuff?

```
                         ;
                         ; SUBROUTINES
                         ;
                         ; BEEP - GENERATE TONE ON KEYBOARD SPEAKER
                         ; ON ENTRY A= FREQ
                         ;
F058 85 40       BEEP    STA   FREQ
F05A A5 14       BEEP1   LDA   RTCLOK+2         CURRENT CLOCK      #25 for PAL
F05C 18                  CLC
F05D 69 1E               ADC   #30     1 SEC TONE
F05F AA                  TAX
F060 A9 FF       WFL     LDA   #$FF
F062 8D 1F D0            STA   CONSOL  TURN ON SPEAKER
F065 A9 00               LDA   #0
F067 A0 F0               LDY   #$F0
F069 88                  DEY
F06A D0 FD               BNE   *-1
F06C 8D 1F D0            STA   CONSOL  TURN OFF SPEAKER
F06F A0 F0               LDY   #$F0
F071 88                  DEY
F072 D0 FD               BNE   *-1
F074 E4 14               CPX   RTCLOK+2         SEE IF 1 SEC IS UP YET
F076 D0 E8               BNE   WFL
F078 C6 40               DEC   FREQ    COUNT BEEPS
F07A F0 0B               BEQ   WFAK    IF ALL DONE GO WAIT FOR KEY
F07C 8A                  TXA
F07D 18                  CLC
F07E 69 0A               ADC   #10 — 08 for PAL
F080 AA                  TAX
F081 E4 14               CPX   RTCLOK+2
F083 D0 FC               BNE   *-2
F085 F0 D3               BEQ   BEEP1   UNCOND GO BEEP AGIN
F087 20 8C F0    WFAK    JSR   WFAK1            USE SIMULATED "JMP (KGETCH)"
F08A 98                  TYA
F08B 60                  RTS
F08C AD 25 E4    WFAK1   LDA   KEYBDV+5
F08F 48                  PHA
F090 AD 24 E4            LDA   KEYBDV+4         SIMULATE "JMP (KGETCH)"
F093 48                  PHA
F094 60                  RTS
                         ;
                         ; SIOSB - CALL SIO ON SYSTEM BUFFER
                         ;
F095 8D 02 03    SIOSB   STA   DCOMND  SAVE COMMAND
F098 A9 00               LDA   #0
F09A 8D 09 03            STA   DBYTHI  SET BUFFER LENGTH
F09D A9 83               LDA   #131
F09F 8D 08 03            STA   DBYTLO
F0A2 A9 03               LDA   #CBUFH
F0A4 8D 05 03            STA   DBUFHI  SET BUFFER ADDRESS
F0A7 A9 FD               LDA   #CBUFL
F0A9 8D 04 03            STA   DBUFLO
F0AC A9 60       CSIO    LDA   #$60    CASSET PSEUDO DEVICE
F0AE 8D 00 03            STA   DDEVIC
F0B1 A9 00               LDA   #0
F0B3 8D 01 03            STA   DUNIT
F0B6 A9 23               LDA   #35     DEVICE TIMEOUT (5/30/79)
F0B8 8D 06 03            STA   DTIMLO
F0BB AD 02 03            LDA   DCOMND  GET COMMAND BACK
F0BE A0 40               LDY   #SRSTA  $40= READ STATUS COMMAND
```

```
F0BE A0 40                    LDY     #SRSTA  SIO READ STATUS COMMAND
```

```
F0C0 C9 52                    CMP     #'R
F0C2 F0 02                    BEQ     *+4
F0C4 A0 80                    LDY     #SWSTA  SIO WRITE STATUS COMMAND
F0C6 8C 03 03                 STY     DSTATS  SET STATUS FOR SIO
F0C9 A5 3E                    LDA     FTYPE
F0CB 8D 0B 03                 STA     DAUX2   INDICATE IF SHORT IRG MODE
F0CE 20 59 E4                 JSR     SIOV    GO CALL SIO
F0D1 60                       RTS
                      ;
                      ; WSIOSB - WRITE SIO SYSTEM BUFFER
                      ;
F0D2 8D FF 03         WSIOSB  STA     CASBUF+2        STORE TYPE BYTE
F0D5 A9 55                    LDA     #$55
F0D7 8D FD 03                 STA     CASBUF+0
F0DA 8D FE 03                 STA     CASBUF+1
F0DD A9 57                    LDA     #'W     WRITE
F0DF 20 95 F0                 JSR     SIOSB   CALL SIO ON SYSTEM BUFFER
F0E2 60                       RTS     AND RETURN
F0E3                  CRNTPC  =*
                              *=$14
0014 00               CASSPR  .BYTE   MONORG-CRNTPC   CASCV IS TOO LONG
                              .END
```

```
                SYMBOL  TABLE
   ADDCOR  00BE     ADRESS  0064      ALT     0300      ALLPOT  D20C
   ANTIC   D400     APPEND  0081      APPMHI  000E      ATACHR  02FB
   ATAN    BE43     ATEOF   F008      ATRACT  004D      AUDC1   D201
   AUDC2   D203     AUDC3   D205      AUDC4   D207      AUDCTL  D208
   AUDF1   D200     AUDF2   D202      AUDF3   D204      AUDF4   D206
   BADIOC  0096     BADMOD  0091      BEEP    F059      BELP1   F05A
   BFENHI  0035     BFENLO  0034      BITMSK  006E      BLIM    028A
   BLKBDV  E471     BOOT?   0009      BOOTAD  0242      BOTSCR  02BF
   BPTR    003D     BRKABT  0080      BRKKEY  0011      BUFADR  0015
   BUFCNT  006B     BUFRFL  0038      BUFRHI  0033      BUFRLO  0032
   BUFSTR  006C     CASBUF  03FD      CASETV  E440      CASFLG  030F
   CASINI  0002     CASORG  EF43      CASSBT  004B      CASSET  0043
   CASSPR  0014     CAUX1   023C      CAUX2   023D      CBAUDH  02EF
   CBAUDL  02EE     CBUFH   0803      CBUFL   00FD      CCOMND  023B
   CDEVIC  023A     CDTMA1  0226      CDTMA2  0228      CDTMF3  022A
   CDTMF4  022C     CDTMF5  022E      CDTMV1  0218      CDTMV2  021A
   CDTMV3  021C     CDTMV4  021E      CDTMV5  0220      CH      02FC
   CH1     02F2     CHACT   02F3      CHACTL  D401      CHAR    02FA
   CHBAS   02F4     CHBASE  D409      CHKERR  008E      CHKSNT  003B
   CHKSUM  0031     CHRORG  E000      CIOCHR  002F      CIOINV  E46E
   CIOORG  E4A6     CIOV    E456      CIX     00F2      CKEY    004A
   CLOSE   000C     CLOSEC  F028      CLWRT   F038      COLAC   0072
   CLRDHK  001A     COLCRS  0055      COLDST  0244      COLDSV  E477
   COLINC  007A     COLOR0  02C4      COLOR1  02C5      COLOR2  02C6
   COLOR3  02C7     COLOR4  02C8      COLPF0  D016      COLPF1  D017
   COLPF2  D018     COLPF3  D019      COLPM0  D012      COLPM1  D013
   COLPM2  D014     COLPM3  D015      COLRSH  004F      CONSOL  D01F
   COS     BD73     COUNTR  007E      CR      009B      CRETRY  0036
   CRITIC  0042     CRNTPC  F0E3      CRSINH  02F0      CRSROR  008D
   CSIO    F0AC     CSOPIV  E47D      CSTAT   0288      CTIA    D000
   DAUX1   030A     DAUX2   030B      DBSECT  0241      DBUFHI  0305
   DBUFLO  0304     DBYTHI  0309      DBYTLO  0308      DCB     0300
   DCOMND  0302     DDEVIC  0300      DEGFLG  00FB      DEGON   0006
   DELETE  0021     DELTAC  0077      DELTAR  0076      DERROR  0090
   DFLAGS  0240     DIGRT   00F1      DINDEX  0057      DIRECT  0002
   DISK    0044     DISKIV  E450      DISPLY  0053      DLISTH  D403
   DLISTL  D402     DMACTL  D400      DMASK   02A0      DNACK   008B
   DOSINI  000C     DOSVEC  000A      DRAWLN  0011      DRETRY  0037
   DRKMSK  004E     DSKFMS  0018      DSKINV  E453      DSKORG  EDEA
   DSKTIM  0246     DSKUTL  001A      DSPFLG  02FE      DSTAT   004C
   DSTATS  0303     DY1     00FA      DTA     00FC      DTIMLO  0306
   DUNIT   0301     DUNUSE  0007      DVSTAT  02EA      EDITRV  E400
   EEXP    00ED     ENDPT   0074      EOFERR  0088      EOL     00FE
   ERRFLG  023F     ESCFLG  02A2      ESIGN   00EF      EXP     DDC0
   EXP10   DDCC     FADD    DA66      FASC    D8E6      FCAX    F032
   FCHRFL  00F0     FDIV    DB28      FEOF    003F      FILDAT  02FD
   FILFLG  02B7     FILLIN  0012      FLD0P   DD8D      FLD0R   DD89
   FLD1P   DD9C     FLD1R   DD98      FLPTR   00FC      FMOVE   DDB6
   FMSZPG  0043     FMUL    DADB      FNCNOT  0092      FORMAT  0022
   FPI     D9D2     FPREC   0006      FPSCR   05E6      FPSCR1  05EC
   FPTR2   00FE     FR0     00D4      FR1     00E0      FR2     00E6
   FRE     00DA     FREQ    0040      FRMERR  008C      FRX     00EC
   FSCR    05E6     FSCR1   05EC      FSTOP   DDAB      FSTOR   DDA7
   FSUB    DA60     FTYPE   00C6      GET     EFE8      GBYTE   EF86
   GETCHR  0007     GETREC  0005      GLBABS  02E0      GPRIOR  026F
   GRACTL  D01D     GRAFM   D011      GRAFP0  D00D      GRAFP1  D00E
   GRAFP2  D00F     GRAFP3  D010      HATABS  031A      HDR     00FB
   HITCLR  D01E     HOLD1   0051      HOLD2   029F      HOLD3   029D
   HOLD4   02BC     HOLD5   02BD      HBLNCH  007C      HPRGNT  000A
```

```
CASSET HANDLER 3/12 (DK1:CASEV)

HPOSM1  D005     HPOSM2  D006     HPOSM3  D007     HPOSP0  D000
HPOSP1  D001     HPOSP2  D002     HPOSP3  D003     HSCROL  D404
ICAX1   034A     ICAX1Z  002A     ICAX2   034B     ICAX2Z  002B
ICBAH   0345     ICBAHZ  0025     ICBAL   0344     ICBALZ  0024
ICBLH   0349     ICBLHZ  0029     ICBLL   0348     ICBLLZ  0028
ICCOM   0342     ICCOMT  0017     ICCOMZ  0022     ICDNO   0341
ICDNOZ  0021     ICHID   0340     ICHIDZ  0020     ICIDNO  002E
ICPTH   0347     ICPTHZ  0027     ICPTL   0346     ICPTLZ  0026
ICSPR   034C     ICSPRZ  002C     ICSTA   0343     ICSTAZ  0023
IFP     D9AA     INBUFF  00F3     INIT    EF41     INSCLR  0070
INSDAT  007D     INTABS  0200     INTEMP  022D     INTINV  E46B
INTORG  E6D5     INTZBS  0010     INVFLG  02B6     IOCB    0340
IOCBAS  0020     IOCBSZ  0010     IOCFRE  00FF     IRQEN   D20E
IRQST   D20E     ISEOF   F00D     KBCODE  D209     KBD     004B
KBDORG  E3E4     KEYBDV  E420     KEYDEL  02F1     KNFEND  05FF
LBPR1   057E     LBPR2   057F     LBUFF   0580     LEDGE   0002
LINBUF  0247     LINZBS  0000     LMARGN  0052     LOCKFL  0023
LOG     DECD     LOG10   DED1     LOGCOL  0063     LOGMAP  02B2
LPENH   0234     LPENV   0235     M0PF    D000     M0PL    D008
M1PF    D001     M1PL    D009     M2PF    D002     M2PL    D00A
M3PF    D003     M3PL    D00B     MAXDEV  0021     MAXIOC  0080
MEMLO   02E7     MEMTOP  02E5     MLTTMP  0066     MODEM   004D
MONORG  F0E3     MOTRGO  0034     MSTRST  003C     MXDMOD  0010
NEWCOL  0061     NEWROW  0060     NMI     F005     NMIEN   D40E
NMIRES  D40F     NMIST   D40F     NOCKSM  003C     NONDEV  0082
NOTE    0026     NOTOPN  0085     NSIGN   00EE     NVALID  0084
OLDADR  005E     OLDCHR  005D     OLDCOL  005B     OLDROW  005A
OPEN    0003     OPENC   EF4C     OPINF   EF5D     OPNIN   0004
OPNINO  000C     OPNOT   0008     OPNRTN  EF8F     OPNTMP  0066
OPOK    EFD3     OPOUT   EF95     OVRRUN  008E     P0PF    D004
P0PL    D00C     P1PF    D005     P1PL    D00D     P2PF    D006
P2PL    D00E     P3PF    D007     P3PL    D00F     PACTL   D302
PADDL0  0270     PADDL1  0271     PADDL2  0272     PADDL3  0273
PADDL4  0274     PADDL5  0275     PADDL6  0276     PADDL7  0277
PBCTL   D303     PBPNT   001D     PBRK    EF8B     PBUFSZ  001E
PBYTE   F010     PCOLR0  02C0     PCOLR1  02C1     PCOLR2  02C2
PCOLR3  02C3     PENH    D40C     PENV    D40D     PIA     D300
PLYARG  05E0     PLYEVL  D040     PMBASE  D407     POINT   0025
POKEY   D200     POKMSK  0010     PORTA   D300     PORTB   D301
POT0    D200     POT1    D201     POT2    D202     POT3    D203
POT4    D204     POT5    D205     POT6    D206     POT7    D207
POTGO   D20B     PRINTR  0050     PRINTV  E430     PRIOR   D01B
PRNBUF  03C0     PRNORG  EE78     PRVOPN  0081     PTEMP   001F
PTIMOT  001C     PTRIG0  027C     PTRIG1  027D     PTRIG2  027E
PTRIG3  027F     PTRIG4  0280     PTRIG5  0281     PTRIG6  0282
PTRIG7  0283     PUTCHR  000B     PUTREC  0009     RADFLG  00FB
RADON   0000     RAMLO   0004     RAMSIZ  02E4     RAMTOP  006A
RANDOM  D20A     RBLOKX  [illegible]     RBLOKV  E47A     RDONLY  0087
RECVDN  0039     REDGE   0027     RENAME  0020     RMARGN  0053
ROWAC   0070     ROWCRS  0054     ROWINC  0079     RTCLOK  0012
SAVADR  0068     SAVIO   0316     SAVMSC  0058     SCREDT  0045
SCRENV  E410     SCRFLG  02BB     SCRMEM  0093     SDLSTH  0231
SDLSTL  0230     SDMCTL  022F     SENDEV  E468     SERIN   D20D
SEROUT  D20D     SETVBV  E45C     SEH     EF64     SHFAMT  006F
SHFLOK  02BE     SIN     BDB1     SIOINV  E465     SIOORG  E944
SIOSB   [illegible]     SIOV    E459     SIZEM   D00C     SIZEP0  D008
SIZEP1  D009     SIZEP2  D00A     SIZEP3  D00B     SKCTL   D20F
SKRES   D20A     SKSTAT  D20F     SOUNDR  0041     SPECIA  EF48
SPECIL  0002     SQR     BEB1     SRSTA   0040     SRTIMR  022B
SSFLAG  02FF     SSKCTL  0232     STACKP  0318     STATUS  0030
```

```
CASSET HANDLER 3/12 (DK1 CASCV)

STATV   F02B    STATUS  0030    STICK0  0278    STICK1  0279
STICK2  027A    STICK3  027B    STIMER  D209    STRIG0  0284
STRIG1  0285    STRIG2  0286    STRIG3  0287    SUBTMP  029E
SUCCES  0001    SWPFLG  007B    SWSTA   0080    SYSVBV  E45F
TABMAP  02A3    TEMP    023E    TEMP1   0312    TEMP2   0314
TEMP3   0315    TIMER1  030C    TIMER2  0310    TIMFLG  0317
TIMOUT  00BA    TINDEX  0293    TMPCHR  0050    TMPCOL  02B9
TMPLBT  02A1    TMPROW  02B8    TMPX1   029C    TONE1   0002
TONE2   0001    TRAMSZ  0006    TRIG0   D010    TRIG1   D011
TRIG2   D012    TRIG3   D013    TRNRCD  0089    TSTAT   0319
TSTDAT  0007    TXTCOL  0291    TXTMSC  0294    TXTOLD  0296
TXTROW  0290    UNLOCK  0024    USAREA  0480    VBREAK  0206
VCOUNT  D40B    VCTABL  E480    VDELAY  D01C    VDSLST  0200
VECTBL  E400    VIMIRQ  0216    VINTER  0204    VKEYBD  0208
VPRCED  0202    VSCROL  D405    VSERIN  020A    VSEROC  020E
VSEROR  020C    VTIMR1  0210    VTIMR2  0212    VTIMR4  0214
VVBLKD  0224    VVBLKI  0222    WAITTM  EF7C    WARMST  0008
WARMSV  E474    WDLR    EFC6    WFAK    F087    WFAKI   F0BC
WFL     F060    WMODE   0289    WRONLY  0083    WSIOSB  F0D2
WSYNC   D40A    WTLR    F046    XITVBV  E462    XMTDON  003A
ZIOCB   0020    ZTBUF   F04A    ZTEMP1  00F5    ZTEMP3  00F9
ZTEMP4  00F7
```

```
                              .TITLE 'MONITR ***** MONITR.SRC ***** 3/9/79 ***** 4:00:00 P.M.'


                                   CONSTANT EQUATES

    0009                      PUTTXT   =      $9                  "PUT TEXT RECORD" CIO COMMAND CODE
    0007                      GETCAR   =      $7                  "GET CHARACTER" CIO COMMAND CODE
    000B                      PUTCAR   =      $B                  "PUT CHARACTER" CIO COMMAND CODE
    0000                      INIMLL   =      $00                 INITIAL MEM LO  LOW BYTE
    0007                      INIMLH   =      $07                 INITIAL MEM LO  HIGH BYTE
    009B                      CR       =      $9B                 ASCII CARRIAGE RETURN
    0001                      GOOD     =      $1                  GOOD STATUS CODE
    0057                      WRITE    =      $57                 WRITE COMMAND
    0052                      READ     =      $52                 READ COMMAND
    0053                      STATC    =      $53                 STATUS COMMAND
    0000                      SEX      =      $0                  SCREEN EDITOR IOCB INDEX
    007D                      CLS      =      $7D                 CLEAR SCREEN CODE
    0092                      CTRLC    =      $92                 KEYBOARD CODE FOR 'CONTROL C'
    0088                      EOF      =      $88                 CASSETTE END OF FILE CODE
    0000                      LIRQ     =      $0                  LONG IRQ TYPE CODE

    0004                      BUFFH    =      CASBUF+3/256
    0000                      BUFFL    =      -256*BUFFH+CASBUF+3      BUFFER POINTER


                              ;THE FOLLOWING EQUATES ARE IN THE CARTRIDGE ADDRESS SPACE
                                   *=WARMSV
              1B F1
    E474 4C 06 90                  JMP    $9006               WARM START VECTOR

                                   *=COLDSV
              25 F1
    E477 4C 00 90                  JMP    $9000               COLD START VECTOR (9000 FOR RAM VECTOR WRITING)

                                   *=$9000
    9000 20 0C 90                  JSR    $900C               (TO HANDLE RAM VECTOR WRITING)
    9003 4C 25 F1                  JMP    PWRUP
    9006 20 0C 90                  JSR    $900C
    9009 4C 1B F1                  JMP    RESET


                                   *= MONORG



                                   HANDLER TABLE ENTRIES

    F0E3 50                   TBLENT  .BYTE   'P'
    F0E4 30 E4                        .WORD   PRINTV
    F0E6 43                           .BYTE   'C'
    F0E7 40 E4                        .WORD   CASETV
    F0E9 45                           .BYTE   'E'
    F0EA 00 E4                        .WORD   EDITRV
    F0EC 53                           .BYTE   'S'
    F0ED 10 E4                        .WORD   SCRENV
```

```
VVBLKD 0224    VVBLKI 0222    WAITTA EF7C    WARMST 0008
[illegible]
[illegible]

MONITOR  *****  MONITR.SRC  *****  2/9/79  *****  4:00:00 P.M.

  F0EF 4B                      .BYTE  'K'
  F0F0 20 E4                   .WORD  KEYBDV

  000E               TBLLEN  =        IDENT-TBLENT-1  HANDLER TABLE LENGTH

                              ***** PRINT MESSAGES *****

  F0F2 7D 41 54      IDENT:   .BYTE CLS,"ATARI COMPUTER - MEMO PAD",CR
  F0F5 41 52 49
  F0F8 20 43 4F
  F0FB 4D 50 55
  F0FE 54 45 52
  F101 20 2D 20
  F104 4D 45 4D
  F107 4F 20 50
  F10A 41 44 9B

  00F0               IDENTH  =        IDENT/256
  00F2               IDENTL  =        -256*IDENTH+IDENT              SYSTEM I.D. MSG POINTER

  F10D 42 4F 4F      DERR:    .BYTE 'BOOT ERROR',CR
  F110 54 20 45
  F113 52 52 4F
  F116 52 9B

  00F1               DERRH   =        DERR/256
  000D               DERRL   =        -256*DERRH+DERR          DISK ERROR MSG POINTER



                          DEVICE/FILENAME SPECIFICATIONS

  F118 45 3A 9B      OPNEDT   .BYTE   "E:",CR           "OPEN SCREEN EDITOR" DEVICE SPEC

  00F1               OPNH    =        OPNEDT/256
  0018               OPNL    =        -256*OPNH+OPNEDT              SCREEN EDITOR OPEN POINTER



                     ;****************************************************************
                     ;         RESET BUTTON ROUTINE STARTS HERE
                     ;****************************************************************

  F11B 78            RESET    SEI                         DISABLE IRQ INTERRUPTS
  F11C AD 44 02               LDA     COLDST              WERE WE IN MIDDLE OF COLDSTART?
  F11F D0 04                  BNE     PWRUP               YES, GO TRY IT AGAIN
  F121 A9 FF                  LDA     #$FF
  F123 D0 03                  BNE     PWRUP1              SET WARM START FLAG


                     ;****************************************************************
                     ;         POWER UP ROUTINES START HERE
                     ;****************************************************************


MONITOR  *****  MONITR.SRC  *****  2/9/79  *****  4:20:02 P.M.
```

```
 F125 78             PWRUP   SEI                        DISABLE IRQ INTERRUPTS
 F126 A9 00                  LDA     #0                 CLEAR WARMSTART FLAG
 F128 85 08          PWRUP1  STA     WARMST
 F12A D8                     CLD                        CLEAR DECIMAL FLAG
 F12B A2 FF                  LDX     #$FF
 F12D 9A                     TXS                        SET STACK POINTER
 F12E 20 3F F2               JSR     SPECL              CARTRIDGE SPECIAL CASE?
 F131 20 81 F2               JSR     HARDI              DO HARDWARE INITIALIZATION
 F134 A5 08                  LDA     WARMST             IS IT WARMSTART?
 F136 D0 28                  BNE     ZOSRAM             YES, ONLY ZERO OS RAM

 F138 A9 00          ZERORM  LDA     #0
 F13A A0 08                  LDY     #WARMST
 F13C 85 04                  STA     RAMLO
 F13E 85 05                  STA     RAMLO+1            INITIALIZE RAM POINTER
 F140 91 04          CLRRAM  STA     (RAMLO),Y          CLEAR MEMORY LOC
 F142 C8                     INY
 F143 C0 00                  CPY     #0                 AT END OF PAGE?
 F145 D0 F9                  BNE     CLRRAM
 F147 E6 05                  INC     RAMLO+1            YES, INCR PAGE POINTER
 F149 A6 05                  LDX     RAMLO+1
 F14B E4 06                  CPX     TRAMSZ             AT END OF MEM?
 F14D D0 F1                  BNE     CLRRAM             NO.

                     ; INITIALIZE DOSVEC TO POINT TO SIGNON (BLACKBOARD)        LDA #SIGNON
 F14F AD 72 E4               LDA     BLKBDV+1
 F152 85 0A                  STA     DOSVEC             USE BLACKBOARD VECTOR       LDA #SIGNON/256
 F154 AD 73 E4               LDA     BLKBDV+2           FOR DOSVEC
 F157 85 0B                  STA     DOSVEC+1
 F159 A9 FF                  LDA     #$FF
 F15B 8D 44 02               STA     COLDST             SET TO SHOW IN MIDDLE OF COLDSTART
 F15E D0 13                  BNE     ESTSCM             GO AROUND ZOSRAM
                     ;
                     ; CLEAR OS RAM (FOR WARMSTART)
 F160 A2 00          ZOSRAM  LDX     #0
 F162 8A                     TXA
 F163 9D 00 02       ZOSRM2  STA     $200,X             CLEAR PAGES 2 AND 3
 F166 9D 00 03               STA     $300,X
 F169 CA                     DEX
 F16A D0 F7                  BNE     ZOSRM2
 F16C A2 10                  LDX     #INTZBS
 F16E 95 00          ZOSRM3  STA     0,X                CLEAR ZERO PAGE LOCATIONS INTZBS-7F
 F170 E8                     INX
 F171 10 FB                  BPL     ZOSRM3
                     ;
                     ; ESTABLISH SCREEN MARGINS
 F173 A9 02          ESTSCM  LDA     #LEDGE
 F175 85 52                  STA     LMARGN
 F177 A9 27                  LDA     #REDGE
 F179 85 53                  STA     RMARGN
                     ;
                     ; MOVE VECTOR TABLE FROM ROM TO RAM
 F17B A2 25          OPSYS   LDX     #$25
 F17D BD 80 E4       MOVVEC  LDA     VCTABL,X           ROM TABLE
 F180 9D 00 02               STA     INTABS,X           TO RAM
 F183 CA                     DEX
 F184 10 F7                  BPL     MOVVEC
 F186 20 94 F2               JSR     CSRAM              DO O.S. RAM SETUP △      (sets CKEY [illegible])
 F189 58                     CLI                        ENABLE IRQ INTERRUPTS
```

```
                            ;
                            ;       LINK HANDLERS
                            ;
F18A A2 0E                          LDX     #TBLLEN
F18C BD E3 F0           NXTENT      LDA     TBLENT,X            READ HANDLER TABLE ENTRY
F18F 9D 1A 03                       STA     HATABS,X            PUT IN TABLE
F192 CA                             DEX
F193 10 F7                          BPL     NXTENT              DONE WITH ALL ENTRIES?
                            ;
                            ;
                            ;
                            ;       INTERROGATE CARTRIDGE ADDR. SPACE TO SEE WHICH CARTRIDGES THERE ARE
                            ;
F195 A2 00                          LDX     #0
F197 86 07                          STX     TSTDAT              CLEAR "B" CART. FLAG
F199 86 06                          STX     TRAMSZ              CLEAR "A" CART. FLAG
F19B AE E4 02                       LDX     RAMSIZ
F19E E0 90                          CPX     #$90                RAM IN "B" CART. SLOT?
F1A0 B0 0A                          BCS     ENDBCK
F1A2 AD FC 9F                       LDA     CART-$2000          NO,
F1A5 D0 05                          BNE     ENDBCK              CART. PLUGGED INTO "B" SLOT?
F1A7 E6 07                          INC     TSTDAT              YES, SET "B" CART. FLAG
F1A9 20 36 F2                       JSR     CBINI               INITIALIZE CARTRIDGE "B"

F1AC AE E4 02           ENDBCK      LDX     RAMSIZ
F1AF E0 B0                          CPX     #$B0                RAM IN "A" CART. SLOT?
F1B1 B0 0A                          BCS     ENDACK
F1B3 AE FC BF                       LDX     CART                NO,
F1B6 D0 05                          BNE     ENDACK              CART. PLUGGED INTO "A" SLOT?
F1B8 E6 06                          INC     TRAMSZ              YES, SET "A" CART. FLAG
F1BA 20 39 F2                       JSR     CAINI               INITIALIZE CARTRIDGE "A"
                            ;
                            ;       OPEN SCREEN EDITOR
                            ;
F1BD A9 03              ENDACK      LDA     #3
F1BF A2 00                          LDX     #SEX
F1C1 9D 42 03                       STA     ICCOM,X             OPEN I/O COMMAND
F1C4 A9 1B                          LDA     #OPNL
F1C6 9D 44 03                       STA     ICBAL,X
F1C9 A9 F1                          LDA     #OPNH
F1CB 9D 45 03                       STA     ICBAH,X             SET BUFFER POINTER TO OPEN SCREEN EDITOR
F1CE A9 0C                          LDA     #$C
F1D0 9D 4A 03                       STA     ICAX1,X             SET UP OPEN FOR INPUT/OUTPUT
F1D3 20 56 E4                       JSR     CIOV                GO TO CIO

F1D6 10 03                          BPL     SCRNOK              BR IF NO ERROR
F1D8 4C 25 F1                       JMP     PWRUP               RETRY PWRUP IF ERROR (SHOULD NEVER HAPPEN!)
F1DB E8                 SCRNOK      INX                         SCREEN OK, SO WAIT FOR VBLANK TO
F1DC D0 FD                          BNE     SCRNOK              BRING UP THE DISPLAY
F1DE C8                             INY
F1DF 10 FA                          BPL     SCRNOK
                            ;
                            ;       DO CASSETTE BOOT
                            ;
F1E1 20 ?? F3                       JSR     CSBOOT              CHECK, BOOT, AND INIT
```

```
 F1E1 20 [illegible]          JSR      CSBOOT                CHECK, BOOT, AND INIT
```

```
MONITOR  ****  MONITR.SRC  ****  3/9/79  ****  4:00:00 P.M.

                         ;   CHECK TO SEE IF EITHER CARTRIDGE WANTS DISK BOOT
   F1E4 A5 06                     LDA      TRAMSZ                CHECK BOTH CARTRIDGES
   F1E6 05 07                     ORA      TSTDAT
   F1E8 F0 12                     BEQ      NOCART                NEITHER CARTRIDGE LIVES
   F1EA A5 06                     LDA      TRAMSZ                "A" CART?
   F1EC F0 03                     BEQ      NOA1                  NO
   F1EE AD FD BF                  LDA      CARTFG                GET CARTRIDGE MODE FLAG
   F1F1 A6 07            NOA1     LDX      TSTDAT                "B" CART?
   F1F3 F0 03                     BEQ      NOB1                  NO
   F1F5 0D FD 9F                  ORA      CARTFG-$2000          ADD OTHER FLAG
   F1F8 29 01            NOB1     AND      #1                    DOES EITHER CART WANT BOOT?
   F1FA F0 03                     BEQ      NOBOOT                NO

                         ;   DO DISK BOOT
   F1FC 20 0F F3         NOCART   JSR      BOOT                  CHECK, BOOT, AND INIT

                         ;   GO TO ONE OF THE CARTRIDGES IF THEY SO DESIRE
   F1FF A9 00            NOBOOT   LDA      #0
   F201 8D 44 02                  STA      COLDST                RESET TO SHOW DONE WITH COLDSTART
   F204 A5 06                     LDA      TRAMSZ                "A" CART?
   F206 F0 0A                     BEQ      NOA2                  NO
   F208 AD FD BF                  LDA      CARTFG                GET CARTRIDGE MODE FLAG
   F20B 29 04                     AND      #4                    DOES IT WANT TO RUN?
   F20D F0 03                     BEQ      NOA2                  NO
   F20F 6C FA BF                  JMP      (CARTCS)              RUN "A" CARTRIDGE
   F212 A5 07            NOA2     LDA      TSTDAT                "B" CART?
   F214 F0 0A                     BEQ      NOCAR2                NO
   F216 AD FD 9F                  LDA      CARTFG-$2000          GET "B" MODE FLAG
   F219 29 04                     AND      #4                    DOES IT WANT TO RUN?
   F21B F0 03                     BEQ      NOCAR2                NO
   F21D 6C FA 9F                  JMP      (CARTCS-$2000)        RUN "B" CARTRIDGE

                         ;   NO CARTRIDGES, OR NEITHER WANTS TO RUN,
                         ;   SO GO TO DOSVEC (DOS, CASSETTE, OR BLACKBOARD)
   F220 6C 0A 00         NOCAR2   JMP      (DOSVEC)

                         ;   PRINT SIGN-ON MESSAGE
   F223 A2 F2            SIGNON   LDX      #IDENTL
   F225 A0 F0                     LDY      #IDENTH
   F227 20 B5 F3                  JSR      PUTLIN                GO PUT SIGN-ON MSG ON SCREEN

                         ;
                         ;
                         ;------   BLACKBOARD ROUTINE
   F22A 20 30 F2         BLKBDR   JSR      BLKB2                 "JSR EGETCH"
   F22D 4C 2A F2                  JMP      BLKBDR                FOREVER
   F230 AD 05 E4         BLKB2    LDA      EDITRV+1              HIGH BYTE
   F233 48                        PHA
   F234 AD 04 E4                  LDA      EDITRV                LOW BYTE
   F237 48                        PHA
   F238 60                        RTS                            SIMULATES "JMP (EDITRV)"
                         ;
                         ;
                         ;   CARTRIDGE INITIALIZATION INDIRECT JUMPS
   F239 6C FE BF         CAINI    JMP      (CARTAD)
   F23C 6C FE 9F         CBINI    JMP      (CARTAD-$2000)
```

SUBROUTINES

```
                               CHECK FOR HOW MUCH RAM & SPECIAL CARTRIDGE CASE
                                   IF SPECIAL CARTRIDGE CASE, DON'T GO BACK -- GO TO CART

    F23F AD FC BF    SPECL   LDA    CART            CHECK FOR RAM OR CART
    F242 D0 12               BNE    ENSPEC          GO IF NOTHING OR MAYBE RAM
    F244 EE FC BF            INC    CART            NOW DO RAM CHECK
    F247 AD FC BF            LDA    CART            IS IT ROM?
    F24A D0 0A               BNE    ENSPEC          NO
    F24C AD FD BF            LDA    CARTFG          YES,
    F24F 29 80               AND    #$80            MASK OFF SPECIAL BIT
    F251 F0 03               BEQ    ENSPEC          BIT SET?
    F253 6C FE BF            JMP    (CARTAD)        YES, GO RUN CARTRIDGE

                               CHECK FOR AMOUNT OF RAM

    F256 CE FC BF    ENSPEC  DEC    CART            RESTORE RAM IF NEEDED
    F259 A9 00               LDA    #0
    F25B 85 05               STA    RAMLO+1
    F25D A9 10               LDA    #$10
    F25F 85 06               STA    TRAMSZ          SET RAM POINTER TO 4K
    F261 A0 00               LDY    #0
    F263 B1 05       HOWMCH  LDA    (RAMLO+1),Y     READ RAM LOCATION
    F265 85 07               STA    TSTDAT          SAVE DATA
    F267 49 FF               EOR    #$FF            INVERT IT
    F269 85 04               STA    RAMLO           SAVE INVERTED DATA
    F26B 91 05               STA    (RAMLO+1),Y     WRITE INVERTED DATA
    F26D B1 05               LDA    (RAMLO+1),Y     READ RAM AGAIN
    F26F C5 04               CMP    RAMLO           IS IT THE INVERTED DATA?
    F271 D0 0D               BNE    ENDRAM
    F273 A5 07               LDA    TSTDAT          YES
    F275 91 05               STA    (RAMLO+1),Y     RESTORE ORIGINAL RAM DATA
    F277 A5 06               LDA    TRAMSZ
    F279 18                  CLC
    F27A 69 10               ADC    #$10
```

BPL

```
    F27C 85 06               STA    TRAMSZ          INCR. RAM POINTER BY 4K
    F27E D0 E3               BNE    HOWMCH          GO FIND HOW MUCH RAM
```

```
   F277 A5 06                 LDA     TRAMSZ
   F279 18                    CLC
   F27A 69 10                 ADC     #$10
```

A06110 ***** MONITR 58C ***** 5/5/79 ***** 4:00:00 P.M.

```
   F27C 85 06                 STA     TRAMSZ          INCR. RAM POINTER BY 4K
   F27E D0 E5                 BNE     HOWMCH          GO FIND HOW MUCH RAM
   F280 60           ENDRAM   RTS
                     ;
                     ;
                     ;
                     ;        HARDWARE INITIALIZATION
                     ;
                     ;
   F281 A9 00        HARDI    LDA     #0
   F283 AA                    TAX
   F284 9D 00 D0     CLRCHP   STA     $D000,X
   F287 9D 00 D4              STA     $D400,X
   F28A 9D 00 D2              STA     $D200,X
   F28D 9D 00 D3              STA     $D300,X
   F290 E8                    INX
   F291 D0 F1                 BNE     CLRCHP
   F293 60                    RTS
                     ;
                     ;
                     ;        O.S. RAM SETUP
                     ;
   F294 C6 11        OSRAM    DEC     BRKKEY          TURN OFF BREAK KEY FLAG
   F296 A5 06                 LDA     TRAMSZ          READ RAM SIZE IN TEMP. REG
   F298 8D E4 02              STA     RAMSIZ          SAVE IT IN RAM SIZE
   F29B 8D E6 02              STA     MEMTOP+1             INIT. MEMTOP ADDR HI BYTE
   F29E A9 00                 LDA     #0
   F2A0 8D E5 02              STA     MEMTOP          INIT. MEMTOP ADDR LO BYTE
   F2A3 A9 00                 LDA     #INIMLL
   F2A5 8D E7 02              STA     MEMLO
   F2A8 A9 07                 LDA     #INIMLH
   F2AA 8D E8 02              STA     MEMLO+1         INITIALIZE MEMLO ADDR VECTOR
   F2AD 20 0C E4              JSR     EDITRV+$C       EDITOR INIT
   F2B0 20 1C E4              JSR     SCRENV+$C       SCREEN INIT
   F2B3 20 2C E4              JSR     KEYBDV+$C       KEYBOARD INIT
   F2B6 20 3C E4              JSR     PRINTV+$C       PRINTER HANDLER INIT
   F2B9 20 4C E4              JSR     CASETV+$C       CASSETTE HANDLER INIT
   F2BC 20 6E E4              JSR     CIOINV          CIO INIT
   F2BF 20 65 E4              JSR     SIOINV          SIO INIT
   F2C2 20 6B E4              JSR     INTINV          INTERRUPT HANDLER INIT
   F2C5 AD 1F D0              LDA     CONSOL
   F2C8 29 01                 AND     #$1
   F2CA D0 02                 BNE     NOKEY           GAME START KEY DEPRESSED?
   F2CC E6 4A                 INC     CKEY            YES, SET KEY FLAG
   F2CE 60           NOKEY    RTS
                     ;
                     ;
                     ;        DO BOOT OF DISK
                     ;
   F2CF A5 08        BOOT     LDA     WARMST
   F2D1 F0 0A                 BEQ     NOWARM          WARM START?
   F2D3 A5 09                 LDA     BOOT?           YES,
   F2D5 29 01                 AND     #1
   F2D7 F0 03                 BEQ     NOINIT          VALID BOOT?
   F2D9 20 [illegible] F3     JSR     DINI            YES, RE-INIT. DOS SOFTWARE
   F2DC 60           NOINIT   RTS
   F2DD A9 0         NOWARM   LDA     #1
   F2DF 8D 0 [illegible] 03   STA     DUNIT           ASSIGN DISK DRIVE NO.
   F2E2 A9 [illegible]        LDA     #STATC
```

```
F2E4 8D 02 03              STA    DCOMND           SET UP STATUS COMMAND
F2E7 20 [illegible]        JSR    STATUS           GO DO DISK STATUS
F2EA 10 01                 BPL    DOBOOT           IS STATUS FROM SIO GOOD?
F2EC 60                    RTS                     NO, GO BACK WITH BAD BOOT STATUS

F2ED A9 00        DOBOOT   LDA    #0
F2EF 8D 0B 03              STA    DAUX2
F2F2 A9 01                 LDA    #1
F2F4 8D 0A 03              STA    DAUX1            SET SECTOR # TO 1
F2F7 A9 00                 LDA    #BUFFL
F2F9 8D 04 03              STA    DBUFLO
F2FC A9 04                 LDA    #BUFFH
F2FE 8D 05 03              STA    DBUFHI           SET UP BUFFER ADDR
F301 20 9D F3     SECT1    JSR    GETSEC           GET SECTOR
F304 10 08                 BPL    ALLSEC           STATUS O.K. ?
F306 20 B1 F3     BADDSK   JSR    DSKRDE           NO, GO PRINT DISK READ ERROR
F309 A5 4A                 LDA    CASSBT
F30B F0 E0                 BEQ    DOBOOT           CASSETTE BOOT?
F30D 60                    RTS                     YES, QUIT
F30E A2 03        ALLSEC   LDX    #3               MOVE 4 BYTES TO: DFLAGS, DBSECT & BOOTAD!
F310 BD 00 04     RDBYTE   LDA    CASBUF+3,X       READ A BUFFER BYTE
F313 9D 40 02              STA    DFLAGS,X         STORE IT
F316 CA                    DEX
F317 10 F7                 BPL    RDBYTE           DONE WITH 4 BYTE TRANSFER ?
F319 AD 42 02              LDA    BOOTAD           YES,
F31C 85 04                 STA    RAMLO
F31E AD 43 02              LDA    BOOTAD+1
F321 85 05                 STA    RAMLO+1          PUT BOOT ADDR INTO Z. PAGE RAM
F323 AD 04 04              LDA    CASBUF+7
F326 85 0C                 STA    DOSINI           ESTABLISH DOS INIT ADDRESS
F328 AD 05 04              LDA    CASBUF+8
F32B 85 0D                 STA    DOSINI+1
F32D A0 7F        MVBUFF   LDY    #$7F             YES, SET BYTE COUNT
F32F B9 00 04     MVNXB    LDA    CASBUF+3,Y
F332 91 04                 STA    (RAMLO),Y        MOVE A BYTE FROM SECTOR BUFFER TO BOOT ADDR.
F334 88                    DEY
F335 10 F8                 BPL    MVNXB            DONE ?
F337 18                    CLC                     YES,
F338 A5 04                 LDA    RAMLO
F33A 69 80                 ADC    #$80
F33C 85 04                 STA    RAMLO
F33E A5 05                 LDA    RAMLO+1
F340 69 00                 ADC    #0
F342 85 05                 STA    RAMLO+1          INCR BOOT LOADER BUFFER POINTER
F344 CE 41 02              DEC    DBSECT           DECR # OF SECTORS.
F347 F0 11                 BEQ    ENBOOT           MORE SECTORS ?
F349 EE 0A 03              INC    DAUX1            YES, INCR SECTOR #
F34C 20 9D F3     SECT2    JSR    GETSEC           GO GET SECTOR
F34F 10 DC                 BPL    MVBUFF           STATUS O.K. ?
F351 20 B1 F3              JSR    DSKRDE           NO, GO PRINT DISK READ ERROR
F354 A5 4A                 LDA    CASSBT
F356 D0 AE                 BNE    BADDSK           IF CASSETTE, QUIT
F358 F0 F2                 BEQ    SECT2            IF DISK, TRY SECTOR AGAIN
F35A A5 4A        ENBOOT   LDA    CASSBT
F35C F0 03                 BEQ    XBOOT            CASSETTE BOOT ?
F35E 20 9D F3              JSR    GETSEC           YES, GET EOF RECORD, BUT DON'T USE IT
F361 20 AC F3     XBOOT    JSR    BLOAD            GO EXECUTE BOOT LOADER
F364 B0 A0                 BCS    BADDSK           IF BAD BOOT, DO IT OVER AGAIN
F366 20 7E F3              JSR    DINI             GO INIT. SOFTWARE
F369 E6 09                 INC    BOOT?            SHOW BOOT SUCCESS
```

why add [illegible] [illegible] ?

```
F36B 60                        RTS
F36C 18               BLOAD    CLC
F36D AD 42 02                  LDA     BOOTAD
F370 69 06                     ADC     #6
F372 85 04                     STA     RAMLO
F374 AD 43 02                  LDA     BOOTAD+1
F377 69 00                     ADC     #0
F379 85 05                     STA     RAMLO+1              PUT START ADDR OF BOOTLOADER INTO RAM
F37B 6C 04 00                  JMP     (RAMLO)
F37E 6C 0C 00         DIN1     JMP     (DOSINI)



                               DISPLAY DISK READ ERROR MSG

F381 A2 00            DSKRDE   LDX     #DERRML
F383 A0 F1                     LDY     #DERRMH



                               PUT LINE ON SCREEN AT PRESENT CURSOR POSITION

                                       X-REG -- LO BYTE, BEGIN ADDR OF LINE
                                       Y-REG -- HI BYTE, BEGIN ADDR OF LINE

F385 8A               PUTLIN   TXA
F386 A2 00                     LDX     #BEX
F388 9D 44 03                  STA     ICBAL,X
F38B 98                        TYA
F38C 9D 45 03                  STA     ICBAH,X              SET UP ADDR OF BEGIN OF LINE
F38F A9 09                     LDA     #PUTTXT
F391 9D 42 03                  STA     ICCOM,X              "PUT TEXT RECORD" COMMAND
F394 A9 FF                     LDA     #$FF
F396 9D 48 03                  STA     ICBLL,X              SET BUFFER LENGTH
F399 20 56 E4                  JSR     CIOV                 PUT LINE ON SCREEN
F39C 60                        RTS



                               GET SECTOR FROM DISK 0

F39D A5 4B            GETSEC   LDA     CASSBT
F39F F0 03                     BEQ     DISKM                CASSETTE BOOT ?
F3A1 4C 7A E4                  JMP     RBLOKV               YES, GO TO READ BLOCK ROUTINE
F3A4 A9 52            DISKM    LDA     #READ
F3A6 8D 02 03                  STA     DCOMND               SET READ SECTOR COMMAND
F3A9 A9 01                     LDA     #1
F3AB 8D 01 03                  STA     DUNIT                SET DRIVE NO. TO DRIVE 0
F3AE 20 53 E4                  JSR     DSKINV               GET SECTOR
F3B1 60                        RTS



                               DO CHECK FOR CASSETTE BOOT & IF SO, DO BOOT

F3B2 A5 08            CSBOOT   LDA     WARMST               WARMSTART?
F3B4 F0 0A                     BEQ     CSBOT2               NO
F3B6 A5 09                     LDA     BOOT?                GET BOOT FLAG
```

```
F3B8 29 02                AND   #2             WAS CASSETTE BOOT SUCCESSFUL?
F3BA F0 03                BEQ   NOCSUC         NO
F3BC 20 E1 F3             JSR   CINT           YES, INIT CASSETTE SOFTWARE
F3BF 60          NOCSUC   RTS

F3C0 A5 4A       CSBOOT   LDA   CKEY
F3C2 F0 1C                BEQ   NOCSBT         "C" KEY FLAG SET ?
F3C4 A9 80                LDA   #$80           YES.
F3C6 85 3E                STA   FTYPE          SET ~~LONG~~ SHORT IRG TYPE /CONTINUOUS!
F3C8 E6 4B                INC   CASSBT         SET CASSETTE BOOT FLAG
F3CA 20 7D E4             JSR   CSOPIV         OPEN CASSETTE FOR INPUT
F3CD 20 01 F3             JSR   SECT1          DO BOOT & INIT
F3D0 A9 00                LDA   #0
F3D2 85 4B                STA   CASSBT         RESET CASSETTE BOOT FLAG
F3D4 85 4A                STA   CKEY           CLEAR KEY FLAG
F3D6 06 09                ASL   BOOT?          SHIFT BOOT FLAG (NOW=2 IF SUCCESS)
F3D8 A5 0C                LDA   DOSINI
F3DA 85 02                STA   CASINI         MOVE INIT ADDRESS FOR CASSETTE
F3DC A5 0D                LDA   DOSINI+1
F3DE 85 03                STA   CASINI+1
F3E0 60          NOCSBT   RTS
                 ;
F3E1 6C 02 00    CINT     JMP   (CASINI)       INIT CASSETTE
                 ;*************************************************************
                 ;
                 ;
                 ;                SPARE BYTE OR MODULE TOO LONG FLAG
                 ;
F3E4             CRNTPC   =     *
                 ;
                          *=    $14
0014 00          MONSPR   .BYTE ABDORG-CRNTPC   MONITR TOO LONG
                 ;
                 .END
```

why is there a CASINI & a DOSINI?
Because both can be executed, though [illegible];
cassette boot is independent of cartridge/disk boot!

```
MONITOR  *****  MONITP SRC  *****  3/9/79  *****  4 00 00 P M

          SYMBOL  TABLE
  ADDCOR  030E     ADRESS  0064     AFP     D800     ALLPOT  D208
  [illegible] F30E ANTIC   D400     APPEND  0001     APPMHI  000E
  ATACHR  02FB     ATAN    BE43     ATRACT  004D     AUDC1   D201
  AUDC2   D203     AUDC3   D205     AUDC4   D207     AUDCTL  D208
  AUDF1   D200     AUDF2   D202     AUDF3   D204     AUDF4   D206
  [illegible] F306 BADIOC  00E6     BADMOD  0091     BFENHI  0035
  BFENLO  0034     BITMSK  006E     BLACKB  F22A     BLIM    028A
  BLKBD   F230     BLKBDV  E471     BLOAD   F36C     BOOT    F2CF
  BOOT?   0009     BOOTAD  02A2     BOTSCR  02BF     BPTR    003D
  BRKABT  0080     BRKKEY  0011     BUFADR  0015     BUFCNT  006B
  [illegible] 0004 BUFFL   000D     BUFRFL  0038     BUFRHI  0033
  BUFRLO  0032     BUFSIZ  006C     CAINI   F239     CART    BFFC
  CARTAD  BFFE     CARTCS  BFFA     CARTFG  BFFD     CASBUF  03FD
  CASETV  E440     CASFLG  030F     CASINI  0002     CASDRV  EF41
  CASTAT  0048     CASSET  0043     CAUX1   023C     CAUX2   023D
  CBAUDH  02EF     CBAUDL  02EE     CBINI   F23C     CCOMND  023E
  CDEVIC  023A     CDTMA1  0226     CDTMA2  0228     CDTMF3  022A
  CDTMF4  022C     CDTMF5  022E     CDTMV1  0218     CDTMV2  021A
  CDTMV3  021C     CDTMV4  021E     CDTMV5  0220     CH      02FC
  CH1     02F2     CHACT   02F3     CHACTL  D401     CHAR    02FA
  CHBAS   02F4     CHBASE  D409     CHKERR  008F     CHKSNT  003B
  CHKSUM  0031     CHRORG  E000     CINA    F3E1     CIOCHR  002F
  CIOINV  E46E     CIOORG  E4A6     CIOV    E456     CIX     00F2
  CKEY    004A     CLOSE   000C     CLRCHP  F284     CLRRAM  F140
  [illegible] 007B COLAC   0072     COLBK   D01A     COLCRS  0055
  COLDST  0244     COLDSV  E477     COLINC  007A     COLOR0  02C4
  COLOR1  02C5     COLOR2  02C6     COLOR3  02C7     COLOR4  02C8
  COLPF0  D016     COLPF1  D017     COLPF2  D018     COLPF3  D019
  COLPM0  D012     COLPM1  D013     COLPM2  D014     COLPM3  D015
  COLRSH  004F     CONSOL  D01F     COS     BD73     COUNTR  007E
  [illegible] 0098 CRETRY  0036     CRITIC  0042     CRNTPC  F3E4
  CRSINH  02F0     CRSRCR  008D     CSBOOT  F3B2     CSBOT2  F3C0
  CSOPIV  E47D     CSTAT   0288     CTIA    D000     CTRLC   0092
  DAUX1   030A     DAUX2   030B     DBSECT  0241     DBUFHI  0305
  DBUFLO  0304     DBYTHI  0309     DBYTLO  0308     DCB     0300
  DCOMND  0302     DDEVIC  0300     DEGFLG  00FB     DEGON   0006
  DELETE  0021     DELTAC  0077     DELTAR  0076     DERR    F100
  [illegible] 00F1 DERRL   000D     DERROR  0090     DFLAGS  0240
  DIGRT   00F1     DINDEX  0057     DINT    F37E     DIRECT  0002
  DISK    0044     DISKIV  E450     DISKM   F3A4     DISPLY  0053
  DLISTH  D403     DLISTL  D402     DMACTL  D400     DMASK   02A0
  [illegible] 00BB DOBOOT  F3ED     DOSINI  000C     DOSVEC  000A
  DRAWLN  0011     DRETRY  0037     DRKMSK  004E     DSAFMS  0018
  DSKINV  E453     DSKORG  EDEA     DSKRDE  F3B1     DSKTIM  0246
  DSKUTL  001A     DSPFLG  02FE     DSTAT   004C     DSTATS  0303
  DTIMLO  0306     DUNIT   0301     DUNUSE  0307     DVSTAT  02EA
  EDITRV  E400     EEXP    00ED     ENBOOT  F39A     ENDACK  F1D0
  ENDNCK  F1AC     ENDPT   0074     ENDRAM  F2A0     ENDREC  F2D6
  EOF     0088     EOFERR  008B     ERRFLG  029F     ESCFLG  02A2
  ESIGN   00EF     ESTSCM  F178     EXP     DDC0     EXP10   DDCC
  FADD    DA66     FASC    D8E6     FCHRFL  00F0     FDIV    DB28
  FEOF    003F     FILDAT  02FD     FILFLG  02B7     FILLIN  0012
  FLD0P   DD8D     FLD0R   DD89     FLD1P   DD9C     FLD1R   DD98
  FLPTR   00FC     FMOVE   DDB6     FMSZPG  0043     FMUL    DADB
  [illegible] 0092 FORMAT  0029     FPI     D9D2     FPREC   0006
  [illegible] 05E6 FPSCR1  05EC     FP9S    00FE     FR0     00D4
  FR1     00E0     FR2     00E6     FRE     00DA     FREQ    0040
  FRMERR  008C     FRX     00EC     FSCR    05E6     FSCR1   05EC
  [illegible] DDAB FSTOR   DDA7     FSUB    DA60     FTYPE   003E
```

```
[illegible] ***** [illegible] ***** [illegible] ***** [illegible]

[illegible] 0007     [illegible] [illegible]   [illegible] [illegible]   [illegible] F240
[illegible] [illegible]  [illegible] [illegible]  [illegible] [illegible]  [illegible] [illegible]
[illegible] D011     [illegible] [illegible]   GRAFP1   D00E     GRAFP2   D00F
[illegible] D015     [illegible] [illegible]   HATABS   [illegible]  HITCLR   D01E
[illegible] [illegible]  HOLD2    [illegible]  [illegible] [illegible]  HOLD4    [illegible]
[illegible] [illegible]  HOLDCH   007C     [illegible] [illegible]  HPOSM0   [illegible]
HPOSM1   D005     HPOSM2   D006     HPOSM3   D007     HPOSP0   D000
HPOSP1   D001     HPOSP2   D002     HPOSP3   D003     HSCROL   D404
ICAUX1   034A     ICAX1Z   002A     ICAX2    034B     ICAX2Z   002B
ICBAH    0345     ICBAHZ   0025     ICBAL    0344     ICBALZ   0024
ICBLH    0349     ICBLHZ   0029     ICBLL    0348     ICBLLZ   0028
ICCOM    0342     ICCOMT   0017     ICCOMZ   0022     ICDNO    0341
ICDNOZ   0021     ICHID    0340     ICHIDZ   0020     ICIDNO   002E
ICPTH    0347     ICPTHZ   0027     ICPTL    0346     ICPTLZ   0026
ICSPR    034C     ICSPRZ   002C     ICSTA    0343     ICSTAZ   0023
IDENT    F0F2     IDENTH   00F0     IDENTL   00F2     IFP      D9AA
INBUFF   00F3     INIMLH   [illegible]  INIMLL   0000     [illegible] [illegible]
INSDAT   007D     INTABS   0200     INTEMP   0220     INTINV   E46D
[illegible] E8D5     [illegible] D010     INVFLG   0286     IOCB     0040
IOCBAS   0020     [illegible] [illegible]  IOCFRE   00FF     IRQEN    D20E
IRQST    D20E     KBCODE   D209     KBD      0048     KBDORG   F3E4
KEYBDV   E420     KEYDEL   02F1     LBFEND   05FF     LBPR1    057E
LBPR2    057F     LBUFF    0580     [illegible] 0082     LINBUF   0247
[illegible] 0000     [illegible] 0000     LMARGN   0052     LOCKFL   [illegible]
[illegible] DECD     LOG10    DED1     LOGCOL   0063     LOGMAP   02B2
LPENH    0234     LPENV    0235     M0PF     0000     M0PL     D008
M1PF     D001     M1PL     D009     M2PF     D002     M2PL     D00A
M3PF     D003     M3PL     D00B     MAXDEV   0021     MAXIOC   0080
MEMLO    02E7     MEMTOP   02E5     MLTTMP   0066     MODEM    0040
[illegible] F0E3     MONSPR   0014     MOVVEC   F170     MVBUFF   F32D
[illegible] F32F     MXDMOD   0010     NEWCOL   0061     NEWROW   0060
NMIEN    D40E     NMIRES   D40F     NMIST    D40F     [illegible] F1FF
[illegible] F212     [illegible] F1FB     NOBOOT   F1FF     NOCAR2   F220
NOCART   F1FC     NOCKSM   003C     NOCSB2   F30F     NOCSBT   F3E0
[illegible] F2D6     NOKEY    F26E     NONDEV   0082     NOTE     0026
NOTOPN   0085     NOWARM   F200     NSIGN    00EE     NVALID   0084
[illegible] F18C     OLDADR   005E     OLDCHR   005D     OLDCOL   005B
OLDROW   005A     OPEN     0003     OPNEDT   F118     OPNH     00F1
OPNIN    0004     OPNINO   000C     OPNL     0018     OPNOT    0008
OPNTMP   0066     OPSYS    F178     OSRAM    F294     OVRRUN   008E
P0PF     D004     P0PL     D00C     P1PF     D005     P1PL     D00D
P2PF     D006     P2PL     D00E     P3PF     D007     P3PL     D00F
PACTL    D302     PADDL0   0270     PADDL1   0271     PADDL2   0272
PADDL3   0273     PADDL4   0274     PADDL5   0275     PADDL6   0276
PADDL7   0277     PBCTL    D303     PBPNT    001D     PBUFSZ   001E
PCOLR0   02C0     PCOLR1   02C1     PCOLR2   02C2     PCOLR3   02C3
PENH     D40C     PENV     D40D     PIA      D300     PLYARG   05E0
PLYEVL   D040     PMBASE   D407     POINT    0025     POKEY    D200
POKMSK   0010     PORTA    D300     PORTB    D301     POT0     D200
POT1     D201     POT2     D202     POT3     D203     POT4     D204
POT5     D205     POT6     D206     POT7     D207     POTGO    D20B
PRINTR   0050     PRINTV   E430     PRIOR    D01B     PRNBUF   03C0
[illegible] EE78     PRVOPN   0081     PTEMP    001F     PTIMOT   001C
PTRIG0   027C     PTRIG1   027D     PTRIG2   027E     PTRIG3   027F
PTRIG4   0280     PTRIG5   0281     PTRIG6   0282     PTRIG7   0283
PUTCAR   0006     PUTCHR   000B     PUTLIN   F385     PUTREC   0009
PUTTXT   0009     PWRUP    F125     PWRUP1   F128     RADFLG   00FB
[illegible] 0000     RAMLO    0004     RAMSIZ   02E4     RAMTOP   006A
RANDOM   D20A     RBLOKV   E47A     RDBYTE   F310     RDONLY   0087
READ     0052     RECVDN   0039     REDGE    0027     RENAME   0020
```

```
MONITOR   *****   MONITF.SRC   ***** 3/9/79   *****   9:00:00 P M

RESET    F118    RMARGN   0053    ROWAC    0070    ROWCRS   0054
ROWINC   0079    RTCLOK   0012    SAVADR   0068    SAVIO    0316
SAVMSC   0058    SCREDT   0045    SCRENV   E410    SCRFLG   02BB
SCRMEM   0093    SCRNOK   F1D8    SDLSTH   0231    SDLSTL   0230
SDMCTL   022F    SECT1    F301    SECTX    F34C    SENDEV   E468
SERIN    D20D    SEROUT   D20D    SETVBV   E45C    SEX      0000
SHFAMT   006F    SHFLOK   02BE    SIGNON   F223    SIN      BD81
SIOINV   E465    SIOORG   E944    SIOV     E459    SIZEM    D00C
SIZEP0   D008    SIZEP1   D009    SIZEP2   D00A    SIZEP3   D00B
SKCTL    D20F    SKRES    D20A    SKSTAT   D20F    SOUNDR   0041
SPECIL   000E    SPECL    F25F    SQR      BEB1    SRTIMR   022B
SSFLAG   02FF    SSKCTL   0232    STACKP   0318    STATC    0053
STATIS   000D    STATUS   0030    STICK0   0278    STICK1   0279
STICK2   027A    STICK3   027B    STIMER   D209    STRIG0   0284
STRIG1   0285    STRIG2   0286    STRIG3   0287    SUBTMP   029E
SUCCES   0001    SWPFLG   0078    SYSVBV   E45F    TABMAP   02A3
TBLFNT   F0E3    TBLLEN   000E    TEMP     023E    TEMP1    0312
TEMP2    0314    TEMP3    0315    TIMER1   030C    TIMER2   0310
TIMFLG   0317    TIMOUT   008A    TINDEX   0293    TMPCHR   0050
TMPCOL   02B9    TMPLBT   02A1    TMPROW   02B8    TMPX1    029C
TRAMSZ   0006    TRIG0    D010    TRIG1    D011    TRIG2    D012
TRIG3    D013    TRNRCD   0089    TSTAT    0319    TSTDAT   0007
TXTCOL   0291    TXTMSC   0294    TXTOLD   0296    TXTROW   0290
UNLOCK   0024    USAREA   0480    VBREAK   0206    VCOUNT   D40B
VCTABL   E480    VDELAY   D01C    VDSLST   0200    VECTBL   E400
VIMIRQ   0216    VINTER   0204    VKEYBD   0208    VPRCED   0202
VSCROL   D405    VSERIN   020A    VSEROC   020E    VSEROR   020C
VTIMR1   0210    VTIMR2   0212    VTIMR4   0214    VVBLKD   0224
VVBLKI   0222    WARMST   0008    WARMSV   E474    WMODE    0289
WRITE    0057    WRONLY   0083    WSYNC    D40A    XBOOT    F361
XITVBV   E462    XMTDON   003A    ZERORM   F138    ZIOCB    0020
ZIRRAM   F160    ZOSRM2   F163    ZOSRM3   F16E    ZTEMP1   00F5
ZTEMP3   00F9    ZTEMP4   00F7
```

```
                                   TITLE  'DISPLAY HANDLER -- 10-30-78 -- DISPLC'

                              ;   HANDLER DEPENDENT EQUATES

        009B                  CR      =       $9B             SYSTEM EOL
        007D                  CLRCOD  =       $7D             CLEAR SCREEN ATASCI CODE
        009F                  CNTL1   =       $9F             POKEY KEY CODE FOR ^1

        0068                  FRMADR  =       SAVADR
        0066                  TOADR   =       MLTTMP
```

```
                        ;
                        ;
                                *=EDITRV
                        ;
                        ; SCREEN EDITOR HANDLER ENTRY POINT
                        ;
E400 F8 F3              EDITOR  WORD    EOPEN-1
E402 33 F6                      WORD    RETUR1-1        (CLOSE)
E404 3D F6                      WORD    EGETCH-1
E406 A3 F6                      WORD    EOUTCH-1
E408 33 F6                      WORD    RETUR1-1        (STATUS)
E40A 3C F6                      WORD    NOFUNC-1        (SPECIAL)
E40C 4C E4 F3                   JMP     PWRONA
E40F 00                         BYTE    0               ROM FILLER BYTE
                        ;
                                *=SCRENV
                        ;
                        ; DISPLAY HANDLER ENTRY POINT
                        ;
E410 F5 F3              DISPLA  WORD    DOPEN-1
E412 33 F6                      WORD    RETUR1-1        (CLOSE)
E414 92 F5                      WORD    GETCH-1
E416 B6 F5                      WORD    OUTCH-1
E418 33 F6                      WORD    RETUR1-1        (STATUS)
E41A FB FC                      WORD    DRAW-1  (SPECIAL)
E41C 4C E4 F3                   JMP     PWRONA
E41F 00                         BYTE    0               ROM FILLER BYTE
                        ;
                                *=KEYBDV
                        ;
                        ; KEYBOARD HANDLER ENTRY POINT
                        ;
E420 33 F6              KBDHND  WORD    RETUR1-1
E422 33 F6                      WORD    RETUR1-1        (CLOSE)
E424 E1 F6                      WORD    KGETCH-1
E426 3C F6                      WORD    NOFUNC-1        (OUTCH)
E428 33 F6                      WORD    RETUR1-1        (STATUS)
E42A 3C F6                      WORD    NOFUNC-1        (SPECIAL)
E42C 4C E4 F3                   JMP     PWRONA
E42F 00                         BYTE    0               ROM FILLER BYTE
                        ;
                        ;
                        ; INTERRUPT VECTOR TABLE ENTRY
                                *=VCTABL-INTABS+VKEYBD
E4B8 BE FF                      WORD    PIRQ            KEYBOARD IRQ INTERRUPT VECTOR
```

```
                                   *=NUDORG

     F3E4 A9 FF          PWRONA  LDA     #$FF
     F3E6 8D FC 02               STA     CH
     F3E9 AD E6 02               LDA     MEMTOP+1
     F3EC 29 F0                  AND     #$F0     INSURE 4K PAGE BOUNDARY
     F3EE 85 6A                  STA     RAMTOP
     F3F0 A9 40                  LDA     #$40     DEFAULT TO UPPER CASE ALPHA AT PWRON
     F3F2 8D BE 02               STA     SHFLOK
     F3F5 60                     RTS              POWER ON COMPLETED
```

redundant -- this value is in RAMSIZ.

```
                          ;
                          ; BEGIN DISPLAY HANDLER OPEN PROCESSING
                          ;
F3F6 A5 2B       DOPEN    LDA   ICAX2Z  GET AUX 2 BYTE
F3F8 29 0F                AND   #$F
F3FA D0 06                BNE   OPNCOM  IF MODE ZERO, CLEAR ICAX1Z
F3FC A5 2A       EOPEN    LDA   ICAX1Z  CLEAR "CLR INHIBIT" AND "MXD MODE" BITS
F3FE 29 0F                AND   #$F
F400 85 2A                STA   ICAX1Z
F402 A9 00                LDA   #0
F404 85 57       OPNCOM   STA   DINDEX
F406 A9 E0                LDA   #$E0    INITIALIZE GLOBAL VBLANK RAM
F408 8D F4 02             STA   CHBAS
F40B A9 02                LDA   #2
F40D 8D F3 02             STA   CHACT
F410 8D 2F 02             STA   SDMCTL  TURN OFF DMA NEXT VBLANK
F413 A9 01                LDA   #SUCCES
F415 85 4C                STA   DSTAT   CLEAR STATUS
F417 A9 C0                LDA   #$C0    DO IRQEN
F419 05 10                ORA   POKMSK
F41B 85 10                STA   POKMSK
F41D 8D 0E D2             STA   IRQEN
F420 A9 00                LDA   #0
F422 8D 93 02             STA   TINDEX  TEXT INDEX MUST ALWAYS BE 0
F425 85 64                STA   ADRESS
F427 85 7B                STA   SWPFLG
F429 8D F0 02             STA   CRSINH  TURN CURSOR ON AT OPEN
F42C A0 0E                LDY   #14     CLEAR TAB STOPS
F42E A9 01                LDA   #1      INIT TAB STOPS TO EVERY 8 CHARACTERS
F430 99 A3 02    CLRTBS   STA   TABMAP,Y
F433 88                   DEY
F434 10 FA                BPL   CLRTBS
F436 A2 04                LDX   #4      LOAD COLOR REGISTERS
F438 BD C1 FE    DOPENB   LDA   COLRTB,X
F43B 9D C4 02             STA   COLOR0,X
F43E CA                   DEX
F43F 10 F7                BPL   DOPENB
F441 A4 6A                LDY   RAMTOP  DO TXTMSC=$2C40 (IF MEMTOP=3000)
F443 88                   DEY
F444 8C 95 02             STY   TXTMSC+1
F447 A9 60                LDA   #$60
F449 8D 94 02             STA   TXTMSC
F44C A6 57                LDX   DINDEX
F44E BD 69 FE             LDA   ANCONV,X CONVERT IT TO ANTIC CODE
F451 D0 04                BNE   DOPENA  IF ZERO, IT IS ILLEGAL
F453 A9 91       OPNERR   LDA   #BADMOD SET ERROR STATUS
F455 85 4C                STA   DSTAT
F457 85 51       DOPENA   STA   HOLD1
F459 A5 6A                LDA   RAMTOP  SET UP AN INDIRECT POINTER
F45B 85 65                STA   ADRESS+1
F45D BC 45 FE             LDY   ALOCAT,X ALLOCATE N BLOCKS OF 40 BYTES
F460 A9 28       DOPEN1   LDA   #40
F462 20 21 F9             JSR   DBSUB
F465 88                   DEY
F466 D0 F8                BNE   DOPEN1
F468 AD 6F 02             LDA   GPRIOR  CLEAR GTIA MODES
F46B 29 3F                AND   #$3F
F46D 85 67                STA   OPNTMP+1
F46F A8                   TAY
```

$2F60

```
F470 E0 08                CPX    #8        TEST IF 320X1
F472 90 17                BCC    NOT8
F474 8A                   TXA              GET 2 LOW BITS
F475 6A                   ROR    A
F476 6A                   ROR    A
F477 6A                   ROR    A
F478 29 C0                AND    #$C0      NOW 2 TOP BITS
F47A 05 67                ORA    OPNTMP+1
F47C A8                   TAY
F47D A9 10                LDA    #16       SUBTRACT 16 MORE FOR PAGE BOUNDARY
F47F 20 21 F9             JSR    DBSUB
F482 E0 0B                CPX    #11       TEST MODE 11
F484 D0 05                BNE    NOT8      IF MODE = 11
F486 A9 06                LDA    #6        PUT GTIA LUM VALUE INTO BACKGROUND REGISTER
F488 8D C8 02             STA    COLOR4
F48B 8C 6F 02     NOT8    STY    GPRIOR    STORE NEW PRIORITY
F48E A5 64                LDA    ADRESS    SAVE MEMORY SCAN COUNTER ADDRESS
F490 85 58                STA    SAVMSC
F492 A5 65                LDA    ADRESS+1
F494 85 59                STA    SAVMSC+1
F496 AD 0B D4     VBWAIT  LDA    VCOUNT    WAIT FOR NEXT VBLANK BEFORE MESSING
F499 C9 7A                CMP    #$7A      WITH THE DISPLAY LIST
F49B D0 F9                BNE    VBWAIT
F49D 20 1F F9             JSR    DBDEC     START PUTTING DISPLAY LIST RIGHT UNDER RAM
F4A0 BD 75 FE             LDA    PAGETB,X  TEST IF DISPLAY LIST WILL BE IN TROUBLE
F4A3 F0 06                BEQ    NOMOD     OF CROSSING A 256 BYTE PAGE BOUNDARY
F4A5 A9 FF                LDA    #$FF      IF SO, DROP DOWN A PAGE
F4A7 85 64                STA    ADRESS
F4A9 C6 65                DEC    ADRESS+1
F4AB A5 64        NOMOD   LDA    ADRESS    SAVE END OF DISPLAY LIST FOR LATER
F4AD 85 68                STA    SAVADR
F4AF A5 65                LDA    ADRESS+1
F4B1 85 69                STA    SAVADR+1
F4B3 20 13 F9             JSR    DBDDEC    (DOUBLE BYTE DOUBLE DECREMENT)
F4B6 A9 41                LDA    #$41      (ANTIC) WAIT FOR VBLANK AND JMP TO TOP
F4B8 20 17 F9             JSR    STORE
F4BB 86 66                STX    OPNTMP
F4BD A9 18                LDA    #24       INITIALIZE BOTSCR
F4BF 8D BF 02             STA    BOTSCR
F4C2 A5 57                LDA    DINDEX    DISALLOW MIXED MODE IF MODE.GE.9 ←— why?
F4C4 C9 09                CMP    #9
F4C6 B0 2D                BCS    NOTMXD
F4C8 A5 2A                LDA    ICAX1Z    TEST MIXED MODE
F4CA 29 10                AND    #$10
F4CC F0 27                BEQ    NOTMXD
F4CE A9 04                LDA    #4
F4D0 8D BF 02             STA    BOTSCR
F4D3 A2 02                LDX    #2        ADD 4 LINES OF TEXT AT BOTTOM OF SCREEN
F4D5 A9 02        DOPEN2  LDA    #2
F4D7 20 17 F9             JSR    STORE
F4DA CA                   DEX
F4DB 10 F8                BPL    DOPEN2
F4DD A4 6A                LDY    RAMTOP    RELOAD MSC FOR TEXT
F4DF 88                   DEY
F4E0 98                   TYA
F4E1 20 17 F9             JSR    STORE
F4E4 A9 60                LDA    #$60
F4E6 20 17 F9             JSR    STORE
F4E9 A9 42                LDA    #$42
F4EB 20 17 F9             JSR    STORE
```

```
F4EE 18                   CLC
F4EF A9 0C                LDA    #MXDMDE-NUMDLE  POINT X AT MIXED MODE TABLE
F4F1 65 66                ADC    OPNTMP
F4F3 85 66                STA    OPNTMP
F4F5 A4 66       NOTMXD   LDY    OPNTMP
F4F7 BE 51 FE             LDX    NUMDLE,Y GET NUMBER OF DISPLAY LIST ENTRIES
F4FA A5 51       DOPEN3   LDA    HOLD1    STORE N DLE'S
F4FC 20 17 F9             JSR    STORE
F4FF CA                   DEX
F500 D0 F8                BNE    DOPEN3
F502 A5 57                LDA    DINDEX   DO THE MESSY 320X1 PROBLEM
F504 C9 08                CMP    #8
F506 90 1C                BCC    DOPEN5
F508 A2 5D                LDX    #93      GET REMAINING NUMBER OF DLE'S
F50A A5 6A                LDA    RAMTOP   RELOAD MEMORY SCAN COUNTER
F50C 38                   SEC
F50D E9 10                SBC    #$10
F50F 20 17 F9             JSR    STORE
F512 A9 00                LDA    #0
F514 20 17 F9             JSR    STORE
F517 A9 4F                LDA    #$4F     (ANTIC) RELOAD MSC CODE
F519 20 17 F9             JSR    STORE
F51C A5 51       DOPEN4   LDA    HOLD1    DO REMAINING DLE'S
F51E 20 17 F9             JSR    STORE
F521 CA                   DEX
F522 D0 F8                BNE    DOPEN4
F524 A5 59       DOPEN5   LDA    SAVMSC+1 POLISH OFF DISPLAY LIST
F526 20 17 F9             JSR    STORE
F529 A5 58                LDA    SAVMSC
F52B 20 17 F9             JSR    STORE
F52E A5 51                LDA    HOLD1
F530 09 40                ORA    #$40
F532 20 17 F9             JSR    STORE
F535 A9 70                LDA    #$70     24 BLANK LINES
F537 20 17 F9             JSR    STORE
F53A A9 70                LDA    #$70
F53C 20 17 F9             JSR    STORE
F53F A5 64                LDA    ADRESS   SAVE DISPLAY-LIST ADDRESS
F541 8D 30 02             STA    SDLSTL
F544 A5 65                LDA    ADRESS+1
F546 8D 31 02             STA    SDLSTL+1
F549 A9 70                LDA    #$70     ADD LAST BLANK LINE ENTRY
F54B 20 17 F9             JSR    STORE    POSITION ADRESS=SDLSTL-1
F54E A5 64                LDA    ADRESS   STORE NEW MEMTOP
F550 8D E5 02             STA    MEMTOP
F553 A5 65                LDA    ADRESS+1
F555 8D E6 02             STA    MEMTOP+1
F558 A5 68                LDA    SAVADR
F55A 85 64                STA    ADRESS
F55C A5 69                LDA    SAVADR+1
F55E 85 65                STA    ADRESS+1
F560 AD 31 02             LDA    SDLSTL+1
F563 20 17 F9             JSR    STORE
F566 AD 30 02             LDA    SDLSTL
F569 20 17 F9             JSR    STORE
F56C A5 4C                LDA    DSTAT    IF ERROR OCURRED ON ALLOCATION, OPEN THE EDITOR
F56E 10 07                BPL    DOPEN8
F570 48                   PHA             SAVE STATUS
F571 20 FC F3             JSR    EOPEN    OPEN THE EDITOR
F574 68                   PLA             RESTORE STATUS
```

```
        F575 A8                    TAY                AND RETURN IT TO CIO
        F576 60                    RTS
        F577 A5 2A         DOPEN   LDA     ICAX1Z     TEST CLEAR INHIBIT BIT
        F579 29 20                 AND     #$20
        F57B D0 0B                 BNE     DOPENA
        F57D 20 B9 F7              JSR     CLRSCR     CLEAR SCREEN
        F580 8D 90 02              STA     TXTROW     AND HOME TEXT CURSOR (AC IS ZERO)
        F583 A5 52                 LDA     LMARGN
        F585 8D 91 02              STA     TXTCOL
        F588 A9 22         DOPENA  LDA     #$22       EVERYTHING ELSE IS SET UP
        F58A 0D 2F 02              ORA     SDMCTL     SO TURN ON DMACTL
        F58D 8D 2F 02              STA     SDMCTL
        F590 4C 21 F6              JMP     RETURN

        F593 20 96 FA      GETCH   JSR     RANGE      GETCH DOES INCRSR, GETPLT DOESN'T
        F596 20 A2 F5              JSR     GETPLT
        F599 20 30 F8              JSR     INATAC     CONVERT INTERNAL CODE TO ATASCII
        F59C 20 D4 F9              JSR     INCRSB
        F59F 4C 34 F6              JMP     RETUR1
        F5A2 20 47 F9      GETPLT  JSR     CONVRT     CONVERT ROW/COLUMN TO ADRESS
        F5A5 B1 64                 LDA     (ADRESS),Y
        F5A7 2D A0 02              AND     DMASK
        F5AA 46 6F         SHIFTD  LSR     SHFAMT     SHIFT DATA DOWN TO LOW BITS
        F5AC B0 03                 BCS     SHIFT1
        F5AE 4A                    LSR     A
        F5AF 10 F9                 BPL     SHIFTD     (UNCONDITIONAL)
        F5B1 8D FA 02      SHIFT1  STA     CHAR
        F5B4 C9 00                 CMP     #0         RESTORE FLAGS ALSO
        F5B6 60                    RTS

        F5B7 8D FB 02      OUTCH   STA     ATACHR                              ?
        F5BA 20 96 FA              JSR     RANGE
                                   JSR     OFFCRS
        F5BD AD FB 02      OUTCHA  LDA     ATACHR     TEST FOR CLEAR SCREEN
        F5C0 C9 7D                 CMP     #CLRCOD
        F5C2 D0 06                 BNE     OUTCHE
        F5C4 20 B9 F7              JSR     CLRSCR
        F5C7 4C 21 F6              JMP     RETURN
        F5CA AD FB 02      OUTCHE  LDA     ATACHR     TEST FOR CARRIAGE RETURN
        F5CD C9 9B                 CMP     #CR
        F5CF D0 06                 BNE     OUTCHB
        F5D1 20 30 FA              JSR     DOCRWS     DO CR
        F5D4 4C 21 F6              JMP     RETURN
        F5D7 20 E0 F5      OUTCHB  JSR     OUTPLT
        F5DA 20 D8 F9              JSR     INCRSR
        F5DD 4C 21 F6              JMP     RETURN

        F5E0 AD FF 02      OUTPLT  LDA     SSFLAG     *****LOOP HERE IF START/STOP FLAG IS NON-0
        F5E3 D0 FB                 BNE     OUTPLT
        F5E5 A2 02                 LDX     #2
        F5E7 B5 54         CRLOOP  LDA     ROWCRS,X SAVE CURSOR LOCATION FOR DRAW LINE TO DRAW FROM
        F5E9 95 5A                 STA     OLDROW,X
        F5EB CA                    DEX
        F5EC 10 F9                 BPL     CRLOOP
        F5EE AD FB 02              LDA     ATACHR     CONVERT ATASCII(ATACHR) TO INTERNAL(CHAR)
        F5F1 A8                    TAY                SAVE ATACHR
```

```
DISPLAY HANDLER -- 10-30-78 -- DISPLC

   F5F2 2A                     ROL     A
   F5F3 2A                     ROL     A
   F5F4 2A                     ROL     A
   F5F5 2A                     ROL     A
   F5F6 29 03                  AND     #3
   F5F8 AA                     TAX             X HAS INDEX INTO ATAINT
   F5F9 98                     TYA             RESTORE ATACHR
   F5FA 29 9F                  AND     #$9F    STRIP OFF COLUMN ADDRESS
   F5FC 1D F6 FE               ORA     ATAINT,X OR IN NEW COLUMN ADDRESS
   F5FF 8D FA 02      OUTCH2   STA     CHAR
   F602 20 47 F9               JSR     CONVRT
   F605 AD FA 02               LDA     CHAR
   F608 46 6F         SHIFTU   LSR     SHFAMT  SHIFT UP TO PROPER POSITION
   F60A B0 04                  BCS     SHIFT2
   F60C 0A                     ASL     A
   F60D 4C 08 F6               JMP     SHIFTU
   F610 2D A0 02      SHIFT2   AND     DMASK
   F613 85 50                  STA     TMPCHR  SAVE SHIFTED DATA
   F615 AD A0 02               LDA     DMASK   INVERT MASK
   F618 49 FF                  EOR     #$FF
   F61A 31 64                  AND     (ADRESS),Y MASK OFF OLD DATA
   F61C 05 50                  ORA     TMPCHR  OR IN NEW DATA
   F61E 91 64                  STA     (ADRESS),Y
   F620 60                     RTS
                      ;
                      ;
   F621 20 A2 F5      RETURN   JSR     SETPLT  DO CURSOR ON THE WAY OUT
   F624 85 5D                  STA     OLDCHR
   F626 A6 57                  LDX     DINDEX  GRAPHICS HAVE INVISIBLE CURSOR
   F628 D0 0A                  BNE     RETUR1
   F62A AE F0 02               LDX     CRSINH  TEST CURSOR INHIBIT
   F62D D0 05                  BNE     RETUR1
   F62F 49 80                  EOR     #$80    TOGGLE MSB
   F631 20 FF F5               JSR     OUTCH2  DISPLAY IT
   F634 A4 4C         RETUR1   LDY     DSTAT           RETURN TO SIO WITH STATUS IN Y
   F636 A9 01                  LDA     #SUCCES
   F638 85 4C                  STA     DSTAT   SET STATUS= SUCCESSFUL COMPLETION
   F63A AD FB 02               LDA     ATACHR  PUT ATACHR IN AC FOR RETURN TO CIO
   F63D 60            NOFUNC   RTS             (NON-EXISTENT FUNCTION RETURN POINT)
                      ;
                      ;
                      ; END OF DISPLAY HANDLER
                      ;
```

```
     F63E 20 83 FC       EGETCH  JSR     SWAP
     F641 20 88 FA               JSR     ERANGE
     F644 A5 6B                  LDA     BUFCNT  ANYTHING IN THE BUFFER?
     F646 D0 34                  BNE     EGETC3  YES
     F648 A5 54                  LDA     ROWCRS  NO, SO SAVE BUFFER START ADDRESS
     F64A 85 6C                  STA     BUFSTR
     F64C A5 55                  LDA     COLCRS
     F64E 85 6D                  STA     BUFSTR+1
     F650 20 E2 F6       EGETC1  JSR     KGETCH  LET'S FILL OUR BUFFER
     F653 84 4C                  STY     DSTAT   SAVE KEYBOARD STATUS
     F655 AD FB 02               LDA     ATACHR  TEST FOR CR
     F658 C9 9B                  CMP     #CR
     F65A F0 12                  BEQ     EGETC2
     F65C 20 AD F6               JSR     DOSS    NO, SO PRINT IT
     F65F 20 83 FC               JSR     SWAP    JSR DOSS DID SWAP SO SWAP BACK
     F662 A5 63                  LDA     LOGCOL  BEEP IF NEARING LOGICAL COL 120
     F664 C9 71                  CMP     #113
     F666 D0 03                  BNE     EGETC6
     F668 20 0A F9               JSR     BELL
     F66B 4C 50 F6       EGETC6  JMP     EGETC1
     F66E 20 E4 FA       EGETC2  JSR     OFFCRS  GET BUFFER COUNT
     F671 20 00 FC               JSR     DOBUFC
     F674 A5 6C                  LDA     BUFSTR  RETURN A CHARACTER
     F676 85 54                  STA     ROWCRS
     F678 A5 6D                  LDA     BUFSTR+1
     F67A 85 55                  STA     COLCRS
     F67C A5 6B          EGETC3  LDA     BUFCNT
     F67E F0 11                  BEQ     EGETC5
     F680 C6 6B          EGETC7  DEC     BUFCNT  AND RETURN TILL BUFCNT=0
     F682 F0 0D                  BEQ     EGETC5
     F684 A5 4C                  LDA     DSTAT   IF ERROR, LOOP ON EGETC7 UNTIL BUFFER IS EMPTIED
     F686 30 F8                  BMI     EGETC7
     F688 20 93 F5               JSR     GETCH
     F68B 8D FB 02               STA     ATACHR
     F68E 4C 83 FC               JMP     SWAP    AND RETURN WITHOUT TURNING CURSOR BACK ON
     F691 20 30 FA       EGETC5  JSR     DOCRWS  DO REAL CARRIAGE RETURN
     F694 A9 9B                  LDA     #CR     AND RETURN EOL
     F696 8D FB 02               STA     ATACHR
     F699 20 21 F6               JSR     RETURN  TURN ON CURSOR THEN SWAP
     F69C 84 4C                  STY     DSTAT   SAVE KEYBOARD STATUS
     F69E 4C 83 FC               JMP     SWAP    AND RETURN THROUGH RETUR1

     F6A1 6C 64 00       JSRIND  JMP     (ADRESS)        JSR TO THIS CAUSES JSR INDIRECT

     F6A4 8D FB 02       EOUTCH  STA     ATACHR  SAVE ATASCII VALUE
     F6A7 20 83 FC               JSR     SWAP
     F6AA 20 98 FA               JSR     ERANGE
     F6AD 20 E4 FA       DOSS    JSR     OFFCRS  TURN OFF CURSOR
     F6B0 20 AD FC               JSR     TSTCTL  TEST FOR CONTROL CHARACTERS (Z=1 IF CTL)
     F6B3 F0 09                  BEQ     EOUTC3
     F6B5 0E A2 02       EOUTC6  ASL     ESCFLG  ESCFLG ONLY WORKS ONCE
     F6B8 20 CA F8               JSR     OUTCHR
     F6BB 4C 83 FC       ERETN   JMP     SWAP    AND RETURN THROUGH RETUR1
     F6BE AD FE 02       EOUTC3  LDA     DSPFLG  DO DSPFLG AND ESCFLG
     F6C1 0D A2 02               ORA     ESCFLG
     F6C4 D0 EF                  BNE     EOUTC6  IF NON-0 DISPLAY RATHER THAN EXECUTE IT
```

```
F6C6 0E A2 02                ASL     ESCFLG
F6C9 E8                      INX             PROCESS CONTROL CHARACTERS
F6CA BD C6 FE                LDA     CNTRLS,X GET DISPLACEMENT INTO ROUTINE
F6CD 85 64                   STA     ADRESS
F6CF BD C7 FE                LDA     CNTRLS+1,X       GET HIGH BYTE
F6D2 85 65                   STA     ADRESS+1
F6D4 20 A1 F6                JSR     JSRIND  DO COMPUTED JSR
F6D7 20 21 F6                JSR     RETURN  DO CURSOR
F6DA 4C 93 FC                JMP     SWAP    ALL DONE SO RETURN THROUGH RETUR1
                     ;
                     ;
                     ;
                     ; END SCREEN EDITOR
                     ;
                     ;
                     ; BEGIN KEYBOARD HANDLER
                     ;
                     ;
                     ;
F6DD A9 FF          KGETC2   LDA     #$FF
F6DF 8D FC 02                STA     CH
F6E2 A5 2A          KGETCH   LDA     ICAX1Z  TEST LSB OF AUX1 FOR SPECIAL EDITOR READ MODE
F6E4 4A                      LSR     A
F6E5 B0 62                   BCS     GETOUT
F6E7 A9 80                   LDA     #BRKABT
F6E9 A6 11                   LDX     BRKKEY  TEST BREAK
F6EB F0 58                   BEQ     K7      IF BREAK, PUT BRKABT IN DSTAT AND CR IN ATACHR
F6ED AD FC 02                LDA     CH
F6F0 C9 FF                   CMP     #$FF
F6F2 F0 EE                   BEQ     KGETCH
F6F4 85 7C                   STA     HOLDCH  SAVE CH FOR SHIFT LOCK PROC
F6F6 A2 FF                   LDX     #$FF    "CLEAR" CH
F6F8 8E FC 02                STX     CH
F6FB 20 D8 FC                JSR     CLICK   DO KEYBOARD AUDIO FEEDBACK (A IS OK)
F6FE AA             KGETC3   TAX             DO ASCCON
F6FF E0 C0                   CPX     #$C0    TEST FOR CTL & SHIFT TOGETHER
F701 90 02                   BCC     ASCCO1
F703 A2 03                   LDX     #3      BAD CODE
F705 BD FE FE       ASCCO1   LDA     ATASCI,X
F708 8D FB 02                STA     ATACHR  DONE
F70B C9 80                   CMP     #$80    DO NULLS
F70D F0 CE                   BEQ     KGETC2
F70F C9 81                   CMP     #$81    CHECK ATARI KEY
F711 D0 0B                   BNE     KGETC1
F713 AD B6 02                LDA     INVFLG
F716 49 80                   EOR     #$80
F718 8D B6 02                STA     INVFLG
F71B 4C DD F6                JMP     KGETC2  DONT RETURN A VALUE
F71E C9 82          KGETC1   CMP     #$82    CAPS/LOWER
F720 D0 07                   BNE     K1
F722 A9 00                   LDA     #0      CLEAR SHFLOK
F724 8D BE 02                STA     SHFLOK
F727 F0 B4                   BEQ     KGETC2
F729 C9 83          K1       CMP     #$83    SHIFT CAPS/LOWER
F72B D0 07                   BNE     K2
F72D A9 40                   LDA     #$40
F72F 8D BE 02                STA     SHFLOK  SHIFT BIT
F732 D0 A9                   BNE     KGETC2
```

```
F734 C9 84       K2      CMP     #$84     CNTL CAPS/LOWER
F736 D0 07               BNE     K3
F738 A9 80               LDA     #$80     CNTL BIT
F73A 8D BE 02            STA     SHFLOK
F73D D0 9E               BNE     KGETC2
F73F C9 85       K3      CMP     #$85     DO EOF
F741 D0 0A               BNE     K6
F743 A9 88               LDA     #EOFERR
F745 85 4C       K7      STA     DSTAT
F747 85 11               STA     BRKKEY   RESTORE BREAK
F749 A9 9B       GETOUT  LDA     #CR      PUT CR IN ATACHR
F74B D0 26               BNE     K8       (UNCONDITIONAL)
F74D A5 7C       K6      LDA     HOLDCH   PROCESS SHIFT LOCKS
F74F C9 40               CMP     #$40     REGULAR SHIFT AND CONTROL TAKE PRESEDENCE
F751 B0 15               BCS     K5       OVER LOCK
F753 AD FB 02            LDA     ATACHR   TEST FOR ALPHA
F756 C9 61               CMP     #$61     LOWER CASE A
F758 90 0E               BCC     K5       NOT ALPHA IF LT
F75A C9 7B               CMP     #$7B     LOWER CASE Z+1
F75C B0 0A               BCS     K5       NOT ALPHA IF GE
F75E AD BE 02            LDA     SHFLOK   DO SHIFT/CONTROL LOCK
F761 F0 05               BEQ     K5       IF NO LOCK, DONT RE-DO IT
F763 05 7C               ORA     HOLDCH
F765 4C FE F6            JMP     KGETC3   DO RETRY
F768 20 8D FC    K5      JSR     TSTCTL   DONT INVERT MSB OF CONTROL CHARACTERS
F76B F0 09               BEQ     K4
F76D AD FB 02            LDA     ATACHR
F770 4D B6 02            EOR     INVFLG
F773 8D FB 02    K8      STA     ATACHR
F776 4C 34 F6    K4      JMP     RETUR1   ALL DONE
```

```
                    ;
                    ; CONTROL CHARACTER PROCESSORS
                    ;
 F779 A9 80         ESCAPE  LDA     #$80    SET ESCAPE FLAG
 F77B 8D A2 02              STA     ESCFLG
 F77E 60                    RTS
 F77F C6 54         CRSRUP  DEC     ROWCRS
 F781 10 06                 BPL     COMRET
 F783 AE BF 02              LDX     BOTSCR  WRAPAROUND
 F786 CA                    DEX
 F787 86 54         UPDNCM  STX     ROWCRS
 F789 4C 5C FC      COMRET  JMP     STRBEG  COLVERT ROW AND COL TO LOGCOL AND RETURN
 F78C E6 54         CRSRDN  INC     ROWCRS
 F78E A5 54                 LDA     ROWCRS
 F790 CD BF 02              CMP     BOTSCR
 F793 90 F4                 BCC     COMRET
 F795 A2 00                 LDX     #0
 F797 F0 EE                 BEQ     UPDNCM  (UNCONDITIONAL)
 F799 C6 55         CRSRLF  DEC     COLCRS
 F79B A5 55                 LDA     COLCRS
 F79D 30 04                 BMI     CRSRL1  (IF LMARGN=0, THIS ELIMINATES PROBLEM CASE)
 F79F C5 52                 CMP     LMARGN
 F7A1 B0 04                 BCS     COMRE1
 F7A3 A5 53         CRSRL1  LDA     RMARGN
 F7A5 85 55         LFRTCM  STA     COLCRS
 F7A7 4C DD F6      COMRE1  JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN
 F7AA E6 55         CRSRRT  INC     COLCRS
 F7AC A5 55                 LDA     COLCRS
 F7AE C5 53                 CMP     RMARGN
 F7B0 90 F5                 BCC     COMRE1
 F7B2 F0 F3                 BEQ     COMRE1  (CAUSE BLE)
 F7B4 A5 52                 LDA     LMARGN
 F7B6 4C A5 F7              JMP     LFRTCM  UNCONDITIONAL TO COMMON STORE
 F7B9 20 F3 FC      CLRSCR  JSR     PUTMSC
 F7BC A0 00                 LDY     #0
 F7BE 98                    TYA             PUT 0 IN THE AC
 F7BF 91 64         CLRSC2  STA     (ADRESS),Y (AC IS ZERO)
 F7C1 C8                    INY
 F7C2 D0 FB                 BNE     CLRSC2
 F7C4 E6 65                 INC     ADRESS+1
 F7C6 A6 65                 LDX     ADRESS+1
 F7C8 E4 6A                 CPX     RAMTOP
 F7CA 90 F3                 BCC     CLRSC2
 F7CC A9 FF                 LDA     #$FF    CLEAN UP LOGICAL LINE BIT MAP
 F7CE 99 B2 02      CLRSC3  STA     LOGMAP,Y (Y IS ZERO AFTER CLRSC2 LOOP)
 F7D1 C8                    INY
 F7D2 C0 04                 CPY     #4
 F7D4 90 F8                 BCC     CLRSC3  SET 4 BYTES
 F7D6 20 E4 FC      HOME    JSR     COLCR   PLACE COLCRS AT LEFT EDGE
 F7D9 85 63                 STA     LOGCOL
 F7DB 85 6D                 STA     BUFSTR+1
 F7DD A9 00                 LDA     #0
 F7DF 85 54                 STA     ROWCRS
 F7E1 85 56                 STA     COLCRS+1
 F7E3 85 6C                 STA     BUFSTR
 F7E5 60                    RTS

 F7E6 A5 63         BS      LDA     LOGCOL  BACKSPACE
 F7E8 C5 52                 CMP     LMARGN
```

```
F7EA F0 21                      BEQ     BS1
F7EC A5 55            BS        LDA     COLCRS  LEFT EDGE?
F7EE C5 52                      CMP     LMARGN
F7F0 D0 03                      BNE     BS3     NO
F7F2 20 73 FC                   JSR     DELTIA  YES, SEE IF LINE SHOULD BE DELETED
F7F5 20 99 F7         BS3       JSR     CRSRLF
F7F8 A5 55                      LDA     COLCRS
F7FA C5 53                      CMP     RMARGN
F7FC D0 07                      BNE     BS2
F7FE A5 54                      LDA     ROWCRS
F800 F0 03                      BEQ     BS2
F802 20 7F F7                   JSR     CRSRUP
F805 A9 20            BS2       LDA     #$20    MAKE BACKSPACE DESTRUCTIVE
F807 8D FB 02                   STA     ATACHR
F80A 20 E0 F5                   JSR     OUTPLT
F80D 4C DD F8         BS1       JMP     DOLCOL  AND RETURN
F810 20 AA F7         TAB       JSR     CRSRRT  BEGIN SEARCH
F813 A5 55                      LDA     COLCRS  TEST FOR NEW LINE
F815 C5 52                      CMP     LMARGN
F817 D0 0A                      BNE     TAB1    NO
F819 20 34 FA                   JSR     DOCR    DO CARRIAGE RETURN
F81C 20 20 FB                   JSR     LOGGET  CHECK IF END OF LOGICAL LINE
F81F 90 02                      BCC     TAB1    NO, CONTINUE
F821 B0 07                      BCS     TAB2    (UNCONDITIONAL)
F823 A5 63            TAB1      LDA     LOGCOL  CHECK FOR TAB STOP
F825 20 28 FB                   JSR     BITGET
F828 90 E6                      BCC     TAB     NO, SO KEEP LOOKING
F82A 4C DD F8         TAB2      JMP     DOLCOL  CONVERT ROW AND COL TO LOGCOL AND RETURN
F82D A5 63            SETTAB    LDA     LOGCOL
F82F 4C 06 FB                   JMP     BITSET  SET BIT IN MAP AND RETURN
F832 A5 63            CLRTAB    LDA     LOGCOL
F834 4C 12 FB                   JMP     BITCLR  CLEAR " " " " "
F837 20 9D FC         INSCHR    JSR     PHACRS
F83A 20 A2 F5                   JSR     GETPLT  GET CHARACTER UNDER CURSOR
F83D 85 7D                      STA     INSDAT
F83F A9 00                      LDA     #0
F841 8D BB 02                   STA     SCRFLG
F844 20 FF F5         INSCH4    JSR     OUTCH2  STORE DATA
F847 A5 63                      LDA     LOGCOL  SAVE LOGCOL: IF AFTER INCRSA LOGCOL IS
F849 48                         PHA             < THAN IT IS NOW, END LOOP
F84A 20 DC F9                   JSR     INCRSA  SPECIAL INCRSR ENTRY POINT
F84D 68                         PLA
F84E C5 63                      CMP     LOGCOL
F850 B0 0C                      BCS     INSCH3  QUIT
F852 A5 7D            INSCH1    LDA     INSDAT  KEEP GOING
F854 48                         PHA
F855 20 A2 F5                   JSR     GETPLT
F858 85 7D                      STA     INSDAT
F85A 68                         PLA
F85B 4C 44 F8                   JMP     INSCH4
F85E 20 A8 FC         INSCH3    JSR     PLACRS
F861 CE BB 02         INSCH6    DEC     SCRFLG
F864 30 04                      BMI     INSCH5  IF SCROLL OCCURRED
F866 C6 54                      DEC     ROWCRS  MOVE CURSOR UP
F868 D0 F7                      BNE     INSCH6  (UNCOND) CONTINUE UNTIL SCRFLG IS MINUS
F86A 4C DD F8         INSCH5    JMP     DOLCOL  CONVERT ROW AND COL TO LOGCOL AND RETURN


F86D 20 9D FC         DELCHR    JSR     PHACRS
F870 20 47 F9         DELCH1    JSR     CONVRT  GET DATA TO THE RIGHT OF THE CURSOR
```

```
F873 A5 64                  LDA     ADRESS
F875 85 68                  STA     SAVADR  SAVE ADRESS TO KNOW WHERE TO PUT DATA
F877 A5 65                  LDA     ADRESS+1
F879 85 69                  STA     SAVADR+1
F87B A5 63                  LDA     LOGCOL
F87D 48                     PHA
F87E 20 D4 F9               JSR     INCRSB  PUT CURSOR OVER NEXT CHARACTER
F881 68                     PLA
F882 C5 63                  CMP     LOGCOL  TEST NEW LOGCOL AGAINST OLD LOGCOL
F884 B0 10                  BCS     DELCH2  IF OLD LE NEW THEN QUIT
F886 A5 54                  LDA     ROWCRS  IS ROW OFF SCREEN?
F888 CD BF 02               CMP     BOTSCR
F88B B0 09                  BCS     DELCH2  YES, SO QUIT
F88D 20 A2 F5               JSR     GETPLT  GET DATA UNDER CURSOR
F890 A0 00                  LDY     #0
F892 91 68                  STA     (SAVADR),Y      PUT IT IN PREVIOUS POSITION
F894 F0 DA                  BEQ     DELCH1  AND LOOP (UNCONDITIONAL)
F896 A0 00          DELCH2  LDY     #0
F898 98                     TYA
F899 91 68                  STA     (SAVADR),Y      CLEAR THE LAST POSITION
F89B 20 68 FC               JSR     DELTIA  TRY TO DELETE A LINE
F89E 20 AB FC               JSR     PLACRS
F8A1 4C DD FB               JMP     DOLCOL  AND RETURN
F8A4 38             INSLIN  SEC             NORMAL INSLIN PUTS "1" INTO BIT MAP
F8A5 20 7B FB       INSLIA  JSR     EXTEND  ENTRY POINT FOR C=0
F8A8 A5 52                  LDA     LMARGN  DO CARRIAGE RETURN (NO LF)
F8AA 85 55                  STA     COLCRS
F8AC 20 47 F9               JSR     CONVRT  GET ADDRESS
F8AF A5 64                  LDA     ADRESS  SET UP TO=40+FROM (FROM = CURSOR)
F8B1 85 68                  STA     FRMADR
F8B3 18                     CLC
F8B4 69 28                  ADC     #40
F8B6 85 66                  STA     TOADR
F8B8 A5 65                  LDA     ADRESS+1
F8BA 85 69                  STA     FRMADR+1
F8BC 69 00                  ADC     #0
F8BE 85 67                  STA     TOADR+1
F8C0 A6 54                  LDX     ROWCRS  SET UP LOOP COUNTER
F8C2 E0 17                  CPX     #23
F8C4 F0 08                  BEQ     INSLI2
F8C6 20 4E FB       INSLI1  JSR     MOVLIN
F8C9 E8                     INX
F8CA E0 17                  CPX     #23
F8CC D0 F8                  BNE     INSLI1
F8CE 20 9E FB       INSLI2  JSR     CLRLIN  CLEAR CURRENT LINE
F8D1 4C DD FB               JMP     DOLCOL  COLVERT ROW AND COL TO LOGCOL AND RETURN
F8D4 20 DD FB       DELLIN  JSR     DOLCOL  GET BEGINNING OF LOG LINE (HOLD1)
F8D7 A4 51          DELLIA  LDY     HOLD1   SQUEEZE BIT MAP
F8D9 84 54                  STY     ROWCRS  PUT CURSOR THERE
F8DB A4 54          DELLIB  LDY     ROWCRS
F8DD 98             DELLI1  TYA
F8DE 38                     SEC
F8DF 20 23 FB               JSR     LO2GET  GET NEXT BIT
F8E2 08                     PHP
F8E3 98                     TYA
F8E4 18                     CLC
F8E5 69 78                  ADC     #120
F8E7 28                     PLP
F8E8 20 04 FB               JSR     BITPUT  WRITE IT OVER PRESENT BIT
F8EB C8                     INY
```

```
F8EC C0 18                CPY    #24
F8EE D0 ED                BNE    DELLI1  LOOP
F8F0 AD B4 02             LDA    LOGMAP+2        SET LSB
F8F3 09 01                ORA    #1
F8F5 8D B4 02             STA    LOGMAP+2
F8F8 A5 52        DELLI2  LDA    LMARGN  DELETE LINE OF DATA USING PART OF SCROLL
F8FA 85 55                STA    COLCRS  CR NO LF
F8FC 20 47 F9             JSR    CONVRT
F8FF 20 87 F8             JSR    SCROL1
F902 20 20 F8             JSR    LOGGET  TEST NEXT LINE FOR CONTINUATION
                                         IS IT A NEW LOG LINE?
F905 90 DA                BCC    DELLIB  NO SO DELETE ANOTHER
F907 4C DD F8             JMP    DOLCOL  YES SO DOLCOL AND RETURN
F90A A0 20        BELL    LDY    #$20
F90C 20 D8 FC     BELL1   JSR    CLICK
F90F 88                   DEY
F910 10 FA                BPL    BELL1
F912 60                   RTS
```

```
                          ;
                          ; ROUTINES
                          ;
                          ;
                          ; DOUBLE BYTE DECREMENT OF INDIRECT POINTER
                          ;    INCLUDING DB SUBTRACT AND DB DOUBLE DECREMENT
    F913 A9 02            DBDDEC  LDA     #2
    F915 D0 0A                    BNE     DBSUB   (UNCONDITIONAL)
                          ;
                          ; STORE DATA INDIRECT AND DECREMENT POINTER
                          ; (PLACED HERE TO SAVE JMP DBDEC AFTER STORE)
    F917 A4 4C            STORE   LDY     DSTAT   RETURN ON ERROR
    F919 30 2B                    BMI     STROK
    F91B A0 00                    LDY     #0
    F91D 91 64            STORE1  STA     (ADRESS),Y
                          ;       JMP     DBDEC   DECREMENT AND RETURN
                          ;
    F91F A9 01            DBDEC   LDA     #1
    F921 8D 9E 02         DBSUB   STA     SUBTMP
    F924 A5 4C                    LDA     DSTAT   RETURN ON ERROR
    F926 30 1E                    BMI     STROK
    F928 A5 64                    LDA     ADRESS
    F92A 38                       SEC
    F92B ED 9E 02                 SBC     SUBTMP
    F92E 85 64                    STA     ADRESS
    F930 B0 02                    BCS     DBSUB1
    F932 C6 65                    DEC     ADRESS+1
    F934 A5 0F            DBSUB1  LDA     APPMHI+1 MAKE SURE NOTHING EVER OVERWRITES APPMHI
    F936 C5 65                    CMP     ADRESS+1
    F938 90 0C                    BCC     STROK   OK
    F93A D0 06                    BNE     STRERR  ERROR
    F93C A5 0E                    LDA     APPMHI
    F93E C5 64                    CMP     ADRESS
    F940 90 04                    BCC     STROK
    F942 A9 93            STRERR  LDA     #SCRMEM SHOW MEM TOO SMALL FOR SCREEN ERROR
    F944 85 4C                    STA     DSTAT
    F946 60               STROK   RTS
                          ;
                          ;
                          ;
                          ; CONVERT ROW/COLUMN CURSOR INTO REAL ADDRESS (FROM SAVMSC ON UP)
                          ;
    F947 A5 54            CONVRT  LDA     ROWCRS  SAVE CURSOR
    F949 48                       PHA
    F94A A5 55                    LDA     COLCRS
    F94C 48                       PHA
    F94D A5 56                    LDA     COLCRS+1
    F94F 48                       PHA
    F950 20 F3 FC                 JSR     PUTMSC
    F953 A5 54                    LDA     ROWCRS  PUT 10*ROWCRS INTO MLTTMP
    F955 85 66                    STA     MLTTMP
    F957 A9 00                    LDA     #0
    F959 85 67                    STA     MLTTMP+1
    F95B A5 66                    LDA     MLTTMP  QUICK X8
    F95D 0A                       ASL     A
    F95E 26 67                    ROL     MLTTMP+1
    F960 85 51                    STA     HOLD1   (SAVE 2X VALUE)
    F962 A4 67                    LDY     MLTTMP+1
```

```
F964 8C 9F 02          STY    HOLD2           ''
F967 0A                ASL    A
F968 26 67             ROL    MLTTMP+1
F96A 0A                ASL    A
F96B 26 67             ROL    MLTTMP+1
F96D 18                CLC             ADD IN 2X
F96E 65 51             ADC    HOLD1
F970 85 66             STA    MLTTMP
F972 A5 67             LDA    MLTTMP+1
F974 6D 9F 02          ADC    HOLD2
F977 85 67             STA    MLTTMP+1
F979 A6 57             LDX    DINDEX  NOW SHIFT MLTTMP LEFT DHLINE TIMES TO FINISH
F97B BC B1 FE          LDY    DHLINE,X MULTIPLY
F97E 88         CONVR1 DEY             LOOP N TIMES
F97F 30 07             BMI    CONVR2
F981 06 66             ASL    MLTTMP
F983 26 67             ROL    MLTTMP+1
F985 4C 7E F9          JMP    CONVR1
F988 BC A5 FE   CONVR2 LDY    DIV2TB,X NOW DIVIDE HCRSR TO ACCOUNT FOR PARTIAL BYTES
F98B A5 55             LDA    COLCRS
F98D A2 07             LDX    #7       * TRICKY *
F98F 88         CONVR3 DEY
F990 30 0A             BMI    CONVR4
F992 CA                DEX
F993 46 56             LSR    COLCRS+1
F995 6A                ROR    A
F996 6E A1 02          ROR    TMPLBT  SAVE LOW BITS FOR MASK
F999 4C 8F F9          JMP    CONVR3
F99C C8         CONVR4 INY            SO Y IS ZERO UPON RETURN FROM THIS ROUTINE
F99D 18                CLC
F99E 65 66             ADC    MLTTMP  ADD SHIFTED COLCRS TO MLTTMP
F9A0 85 66             STA    MLTTMP
F9A2 90 02             BCC    CONVR5
F9A4 E6 67             INC    MLTTMP+1
F9A6 38         CONVR5 SEC             * TRICKY *
F9A7 6E A1 02   CONVR6 ROR    TMPLBT  SLIDE A "1" UP AGAINST LOW BITS (CONTINUE TILL X=-1)
F9AA 18                CLC
F9AB CA                DEX            AND FINISH SHIFT SO LOW BITS ARE
F9AC 10 F9             BPL    CONVR6  RIGHT JUSTIFIED.
F9AE AE A1 02          LDX    TMPLBT  TMPLBT IS NOW THE INDEX INTO DMASKTB
F9B1 A5 66             LDA    MLTTMP  PREPARE FOR RETURN
F9B3 18                CLC
F9B4 65 64             ADC    ADRESS
F9B6 85 64             STA    ADRESS
F9B8 85 5E             STA    OLDADR  REMEMBER THIS ADDRESS FOR CURSOR
F9BA A5 65             LDA    MLTTMP+1
F9BC 65 65             ADC    ADRESS+1
F9BE 85 65             STA    ADRESS+1
F9C0 85 5F             STA    OLDADR+1
F9C2 BD B1 FE          LDA    DMASKT,X
F9C5 8D A0 02          STA    DMASK
F9C8 85 6F             STA    SHFAMT
F9CA 68                PLA
F9CB 85 56             STA    COLCRS+1
F9CD 68                PLA
F9CE 85 55             STA    COLCRS
F9D0 68                PLA
F9D1 85 54             STA    ROWCRS
F9D3 60                RTS
```

```
                    ; INCREMENT CURSOR AND DETECT BOTH END OF LINE AND END OF SCREEN

F9D4 A9 00          INCRSB  LDA     #0       NON-EXTEND ENTRY POINT
F9D6 F0 02                  BEQ     INCRSC
F9D8 A9 9B          INCRSR  LDA     #$9B     SPECIAL CASE ELIMINATOR
F9DA 85 7D          INCRSC  STA     INSDAT
F9DC E6 63          INCRSA  INC     LOGCOL   (INSCHR ENTRY POINT)
F9DE E6 55                  INC     COLCRS
F9E0 D0 02                  BNE     INCRS2   DO HIGH BYTE
F9E2 E6 56                  INC     COLCRS+1
F9E4 A5 55          INCRS2  LDA     COLCRS   TEST END OF LINE
F9E6 A6 57                  LDX     DINDEX
F9E8 DD 8D FE               CMP     COLUMN,X          TEST TABLED VALUE FOR ALL SCREEN MODES
F9EB F0 0B                  BEQ     INC2A    DO CR IF EQUAL
F9ED E0 00                  CPX     #0       MODE 0?
F9EF D0 06                  BNE     INCRS3   IF NOT, JUST RETURN
F9F1 C5 53                  CMP     RMARGN   TEST AGAINST RMARGN
F9F3 F0 02                  BEQ     INCRS3   EQUAL IS OK
F9F5 B0 01                  BCS     INC2A    IF GREATER THAN, DO CR
F9F7 60             INCRS3  RTS
F9F8 E0 08          INC2A   CPX     #8                CHECK MODE
F9FA 90 04                  BCC     DOCR1             NOT 320X1 SO DO IT
F9FC A5 56                  LDA     COLCRS+1          TEST MSB
F9FE F0 F7                  BEQ     INCRS3            ONLY AT 64 SO DON'T DO IT
FA00 A5 57          DOCR1   LDA     DINDEX   DONT MESS WITH LOGMAP IF NO MODE ZERO
FA02 D0 30                  BNE     DOCR
FA04 A5 63                  LDA     LOGCOL   TEST LINE OVERRUN
FA06 C9 51                  CMP     #81
FA08 90 0A                  BCC     DOCR1B   IF LESS THAN 81 IT IS DEFINITELY NOT LINE 3
FA0A A5 7D                  LDA     INSDAT
FA0C F0 26                  BEQ     DOCR     ONLY DO LOG LINE OVERFLOW IF INSDAT <>0
FA0E 20 30 FA               JSR     DOCRWS   LOG LINE OVERFLOW IS SPECIAL CASE
FA11 4C 77 FA               JMP     INCRS1   RETURN
FA14 20 34 FA       DOCR1B  JSR     DOCR     GET IT OVER WITH
FA17 A5 54                  LDA     ROWCRS
FA19 18                     CLC              TEST LOGICAL LINE BIT MAP
FA1A 69 78                  ADC     #120
FA1C 20 25 FB               JSR     BITGET
FA1F 90 08                  BCC     DOCR1A   DONT EXTEND IF OVERRUN IS INTO MIDDLE OF LOG LINE
FA21 A5 7D                  LDA     INSDAT   DONT EXTEND IF INSDAT IS ZERO
FA23 F0 04                  BEQ     DOCR1A   (INSCHR SPECIAL CASE)
FA25 18                     CLC              INSERT "0" INTO BIT MAP
FA26 20 A5 FB               JSR     INSLIN
FA29 4C DD FB       DOCR1A  JMP     DOLCOL   CONVERT ROW AND COL TO LOGCOL AND RETURN
FA2C A9 00          NOSCRL  LDA     #0       DOCR WITHOUT SCROLL
FA2E F0 02                  BEQ     NOSCR1   (UNCONDITIONAL)
FA30 A9 9B          DOCRWS  LDA     #$9B     DOCR WITH SCROLLING (NORMAL MODE)
FA32 85 7D          NOSCR1  STA     INSDAT
FA34 20 E4 FC       DOCR    JSR     COLCR    PLACE COLCRS AT LEFT EDGE
FA37 A9 00                  LDA     #0
FA39 85 56                  STA     COLCRS+1
FA3B E6 54                  INC     ROWCRS
FA3D A6 57          DOCR2   LDX     DINDEX
FA3F A0 18                  LDY     #24      SET UP SCROLL LOOP COUNTER
FA41 24 7B                  BIT     SWPFLG
FA43 10 05                  BPL     DOCR2A   BRANCH IF NORMAL
FA45 A0 04                  LDY     #4
FA47 98                     TYA
FA48 D0 03                  BNE     DOCR2B   (UNCONDITIONAL)
```

```
FA4A BD 99 FE     DOCROW  LDA    NOROWS,X GET NO OF ROWS
FA4D C5 54        DOCROW  CMP    ROWCRS
FA4F D0 26                BNE    INCRS1
FA51 8C 9D 02             STY    HOLD3
FA54 8A                   TXA             DONT SCROLL IF MODE <> 0
FA55 D0 20                BNE    INCRS1
FA57 A5 7D                LDA    INSDAT   OR IF INSDAT = 0
FA59 F0 1C                BEQ    INCRS1
                          LDA    INSDAT   IF INSDAT <> $9B THEN ROLL IN A 0
FA5B C9 9B                CMP    #$9B     TO EXTEND BOTTOM LOGICAL LINE
FA5D 38                   SEC
FA5E F0 01                BEQ    DOCR4B
FA60 18                   CLC
FA61 20 AC F9     DOCR4B  JSR    SCROLL   LOOP BACK TO HERE IF >1 SCROLLS
FA64 EE BB 02             INC    SCRFLG
FA67 C6 6C                DEC    BUFSTR   ROWS MOVE UP SO BUFSTR SHOULD TOO
FA69 CE 9D 02             DEC    HOLD3
FA6C AD B2 02             LDA    LOGMAP
FA6F 38                   SEC             FOR PARTIAL LINES: ROLL IN A '1'
FA70 10 EF                BPL    DOCR4B   AGAIN IF PARTIAL LOGICAL LINE
FA72 AD 9D 02             LDA    HOLD3    PLACE CURSOR AT NEW LINE NEAR THE BOTTOM
FA75 85 54                STA    ROWCRS
FA77 4C DD F8     INCRS1  JMP    COLCOL   COLVERT ROW AND COL TO LOGCOL AND RETURN

                  ;
                  ;
                  ; SUBEND: SUBTRACT ENDPT FROM ROWAC OR COLAC. (X=0 OR 2)
                  ;
FA7A 38           SUBEND  SEC
FA7B B5 70                LDA    ROWAC,X
FA7D E5 74                SBC    ENDPT
FA7F 95 70                STA    ROWAC,X
FA81 B5 71                LDA    ROWAC+1,X
FA83 E5 75                SBC    ENDPT+1
FA85 95 71                STA    ROWAC+1,X
FA87 60                   RTS

                  ;
                  ;
                  ; RANGE: DO CURSOR RANGE TEST. IF ERROR, POP STACK TWICE AND JMP RETURN
                  ;        (ERANGE IS EDITOR ENTRY POINT AND TEST IF EDITOR IS OPEN.
                  ;         IF IT ISNT IT OPENS THE EDITOR AND CONTINUES)
                  ;
FA88 AD BF 02     ERANGE  LDA    BOTSCR   IF BOTSCR=4
FA8B C9 04                CMP    #4
FA8D F0 07                BEQ    RANGE    THEN IT IS IN MIXED MODE AND OK
FA8F A5 57                LDA    DINDEX   IF MODE = 0
FA91 F0 03                BEQ    RANGE    THEN IT IS IN EDITOR MODE AND OK
FA93 20 FC F3             JSR    EOPEN    IF NOT, OPEN EDITOR
FA96 A9 27        RANGE   LDA    #39      ***** RANGE CHECK RMARGN ***** SET UP AC
FA98 C5 53                CMP    RMARGN   ***** RANGE CHECK RMARGN ***** COMPARE
FA9A B0 02                BCS    RANGE3   ***** RANGE CHECK RMARGN ***** BRANCH GE
FA9C 85 53                STA    RMARGN   ***** RANGE CHECK RMARGN ***** BAD SO STORE 39
FA9E A6 57        RANGE3  LDX    DINDEX
FAA0 BD 99 FE             LDA    NOROWS,X         CHECK ROWS
FAA3 C5 54                CMP    ROWCRS
FAA5 90 2A                BCC    RNGERR           (ERROR IF TABLE GE ROWCRS)
FAA7 F0 28                BEQ    RNGERR
FAA9 E0 08                CPX    #8       CHECK FOR 320X1
FAAB D0 04                BNE    RANGE1   SPECIAL CASE IT
FAAD A5 56                LDA    COLCRS+1
FAAF F0 13                BEQ    RNGOK    IF HIGH BYTE IS 0, COL IS OK
```

```
                             LDA    COLCRS+1
FAAF F0 13                   BEQ    RNGOK   IF HIGH BYTE IS 0, COL IS OK
```

```
FAB1 C9 01                   CMP    #1
FAB3 D0 1C                   BNE    RNGERR  IF >1, BAD
FAB5 F0 04                   BEQ    RANGE2  IF 1, GO CHECK LOW BYTE
FAB7 A5 56          RANGE1   LDA    COLCRS+1 FOR OTHERS, NON-ZERO HIGH BYTE IS BAD
FAB9 D0 16                   BNE    RNGERR
FABB BD BD FE       RANGE2   LDA    COLUMN,X CHECK LOW BYTE
FABE C5 55                   CMP    COLCRS
FAC0 90 0F                   BCC    RNGERR
FAC2 F0 0D                   BEQ    RNGERR
FAC4 A9 01          RNGOK    LDA    #SUCCES SET STATUS OK
FAC6 85 4C                   STA    DSTAT
FAC8 A9 80                   LDA    #BRKABT PREPARE BREAK ABORT STATUS
FACA A6 11                   LDX    BRKKEY  CHECK BREAK KEY FLAG
FACC 85 11                   STA    BRKKEY  'CLEAR' BREAK
FACE F0 06                   BEQ    RNGER2  IF BREAK, QUIT IMMEDIATELY AND RETURN TO CIO
FAD0 60                      RTS
FAD1 20 D6 F7       RNGERR   JSR    HOME    ON RANGE ERROR, BRING CURSOR BACK
FAD4 A9 8D                   LDA    #CRSROR SHOW CURSOR OVERRANGE ERROR
FAD6 85 4C          RNGER2   STA    DSTAT
FAD8 68             RNGER1   PLA            RESTORE STACK (THIS ROUTINE IS ALWAYS 1 LEVEL
FAD9 68                      PLA            AWAY FROM RETURN TO CIO)
FADA A5 7B                   LDA    SWPFLG  IF SWAPPED, SWAP BACK
FADC 10 03                   BPL    RETUR3
FADE 20 89 FC                JSR    SWAPA   AND DONT DO RETUR1
FAE1 4C 34 F6       RETUR3   JMP    RETUR1  RETURN TO CIO
                    ;
                    ;
                    ; OFFCRS: RESTORE OLD DATA UNDER CURSOR SO IT CAN BE MOVED
                    ;
FAE4 A0 00          OFFCRS   LDY    #0
FAE6 A5 5D                   LDA    OLDCHR
FAE8 91 5E                   STA    (OLDADR),Y
FAEA 60                      RTS
                    ;
                    ;
                    ;
                    ; BITMAP ROUTINES:
                    ;
                    ; BITCON: PUT MASK IN BITMSK AND INDEX IN X
                    ; BITPUT: PUT CARRY INTO BITMAP
                    ; BITROL: ROL CARRY INTO BOTTOM OF BITMAP (SCROLL)
                    ; BITSET: SET PROPER BIT
                    ; BITCLR: CLEAR PROPER BIT
                    ; BITGET: RETURN CARRY SET IF BIT IS THERE
                    ; LOGGET: DO BITGET FOR LOGMAP INSTEAD OF TABMAP
                    ;
FAEB 48             BITCON   PHA
FAEC 29 07                   AND    #7
FAEE AA                      TAX            GET MASK
FAEF BD B5 FE                LDA    MASKTB,X
FAF2 85 6E                   STA    BITMSK
FAF4 68                      PLA            PROCESS INDEX
FAF5 4A                      LSR    A
FAF6 4A                      LSR    A
FAF7 4A                      LSR    A
FAF8 AA                      TAX
FAF9 60                      RTS
```

```
    FAFA 2E B4 02      BITROL  ROL    LOGMAP+2
    FAFD 2E B3 02              ROL    LOGMAP+1
    FB00 2E B2 02              ROL    LOGMAP
    FB03 60                    RTS

    FB04 90 0C         BITPUT  BCC    BITCLR  AND RETURN
                                              OTHERWISE FALL THROUGH TO BITSET AND RETURN
    FB06 20 EB FA      BITSET  JSR    BITCON
    FB09 BD A3 02              LDA    TABMAP,X
    FB0C 05 6E                 ORA    BITMSK
    FB0E 9D A3 02              STA    TABMAP,X
    FB11 60                    RTS

    FB12 20 EB FA      BITCLR  JSR    BITCON
    FB15 A5 6E                 LDA    BITMSK
    FB17 49 FF                 EOR    #$FF
    FB19 3D A3 02              AND    TABMAP,X
    FB1C 9D A3 02              STA    TABMAP,X
    FB1F 60                    RTS

    FB20 A5 54         LOGGET  LDA    ROWCRS
    FB22 18            LO1GET  CLC
    FB23 69 78         LO2GET  ADC    #120  ← =(LOGMAP-TABMAP)*8
    FB25 20 EB FA      BITGET  JSR    BITCON
    FB28 18                    CLC
    FB29 BD A3 02              LDA    TABMAP,X
    FB2C 25 6E                 AND    BITMSK
    FB2E F0 01                 BEQ    BITGE1
    FB30 38                    SEC
    FB31 60            BITGE1  RTS



                       ; INATAC  INTERNAL(CHAR) TO ATASCII(ATACHR) CONVERSION

    FB32 AD FA 02      INATAC  LDA    CHAR
    FB35 A4 57                 LDY    DINDEX  IF GRAPHICS MODES
    FB37 C0 03                 CPY    #3
    FB39 B0 0F                 BCS    INATA1  THEN DONT CHANGE CHAR
    FB3B 2A                    ROL    A
    FB3C 2A                    ROL    A
    FB3D 2A                    ROL    A
    FB3E 2A                    ROL    A
    FB3F 29 03                 AND    #3
    FB41 AA                    TAX
    FB42 AD FA 02              LDA    CHAR
    FB45 29 9F                 AND    #$9F
    FB47 1D F9 FE              ORA    INTATA,X
    FB4A 8D FB 02      INATA1  STA    ATACHR
    FB4D 60                    RTS



                       ; MOVLIN  MOVE 40 BYTES AT FRMADR TO TOADR SAVING OLD TOADR
                       ;         DATA IN THE LINBUF.  THEN MAKE NEXT FRMADR
                       ;         BE AT LINBUF FOR NEXT TRANSFER & TOADR=TOADR+40

    FB4E A9 02         MOVLIN  LDA    #LINBUF/256    SET UP ADRESS=LINBUF$=247
```

```
                                 LDA     #LINBUF/256     SET UP ADRESS=LINBUF+#247

 F850 85 65                      STA     ADRESS+1
 F852 A9 47                      LDA     #LINBUF
 F854 85 64                      STA     ADRESS
 F856 A0 27                      LDY     #39
 F858 B1 66             MOVLI1   LDA     (TOADR),Y        SAVE TO DATA
 F85A 85 50                      STA     TMPCHR
 F85C B1 68                      LDA     (FRMADR),Y       STORE DATA
 F85E 91 66                      STA     (TOADR),Y
 F860 A5 50                      LDA     TMPCHR
 F862 91 64                      STA     (ADRESS),Y
 F864 88                         DEY
 F865 10 F1                      BPL     MOVLI1
 F867 A5 65                      LDA     ADRESS+1         SET UP FRMADR=LAST LINE
 F869 85 69                      STA     FRMADR+1
 F86B A5 64                      LDA     ADRESS
 F86D 85 68                      STA     FRMADR
 F86F 18                         CLC              ADD 40 TO TOADR
 F870 A5 66                      LDA     TOADR
 F872 69 28                      ADC     #40
 F874 85 66                      STA     TOADR
 F876 90 02                      BCC     MOVLI2
 F878 E6 67                      INC     TOADR+1
 F87A 60                MOVLI2   RTS
                     ;
                     ;
                     ;
                     ; EXTEND  EXTEND BIT MAP FROM ROWCRS (EXTEND LOGICAL LINE
                     ;
 F87B 08                EXTEND   PHP              SAVE CARRY
 F87C A0 17                      LDY     #23
 F87E 98                EXTEN1   TYA
 F87F 20 22 FE                   JSR     LO1GET
 F882 08                         PHP
 F883 98                         TYA
 F884 18                         CLC
 F885 69 79                      ADC     #121
 F887 28                         PLP
 F888 20 04 FE                   JSR     BITPUT
 F88B 88                EXTEN3   DEY
 F88C 30 04                      BMI     EXTEN4
 F88E C4 54                      CPY     ROWCRS
 F890 B0 EC                      BCS     EXTEN1
 F892 A5 54             EXTEN4   LDA     ROWCRS
 F894 18                         CLC
 F895 69 78                      ADC     #120
 F897 28                         PLP
 F898 4C 04 FE                   JMP     BITPUT  STORE NEW LINE'S BIT AND RETURN
                     ;
                     ;
                     ;
                     ; CLRLIN  CLEAR LINE CURSOR IS ON
                     ;
 F89B A5 52             CLRLIN   LDA     LMARGN
 F89D 85 55                      STA     COLCRS
 F89F 20 87 F9                   JSR     CONVRT
 F8A2 A0 27                      LDY     #39
 F8A4 A9 00                      LDA     #0
 F8A6 91 64             CLRLI1   STA     (ADRESS),Y
 F8A8 88                         DEY
 F8A9 10 FB                      BPL     CLRLI1
```

```
  FBAB 60                  RTS

                     ;
                     ;
                     ; SCROLL: SCROLL SCREEN
                     ;
  FBAC 20 FA FA      SCROLL  JSR     BITROL  ROLL IN CARRY
  FBAF A5 58                 LDA     SAVMSC  SET UP WORKING REGISTERS
  FBB1 85 64                 STA     ADRESS
  FBB3 A5 59                 LDA     SAVMSC+1
  FBB5 85 65                 STA     ADRESS+1
  FBB7 A0 28         SCROL1  LDY     #40     LOOP
  FBB9 B1 64                 LDA     (ADRESS),Y
  FBBB A6 6A                 LDX     RAMTOP  TEST FOR LAST LINE
  FBBD CA                    DEX
  FBBE E4 65                 CPX     ADRESS+1
  FBC0 D0 08                 BNE     SCROL2
  FBC2 A2 D7                 LDX     #$D7
  FBC4 E4 64                 CPX     ADRESS
  FBC6 B0 02                 BCS     SCROL2
  FBC8 A9 00                 LDA     #0      YES SO STORE ZERO DATA FOR THIS ENTIRE LINE
  FBCA A0 00         SCROL2  LDY     #0
  FBCC 91 64                 STA     (ADRESS),Y
  FBCE E6 64                 INC     ADRESS
  FBD0 D0 E5                 BNE     SCROL1
  FBD2 E6 65                 INC     ADRESS+1
  FBD4 A5 65                 LDA     ADRESS+1
  FBD6 C5 6A             No  CMP     RAMTOP
  FBD8 D0 DD                 BNE     SCROL1
  FBDA 4C DD FB              JMP     DOLCOL  AND RETURN
                     ;
                     ;
                     ; DOLCOL: DO LOGICAL COLUMN FROM BITMAP AND COLCRS
                     ;
  FBDD A9 00         DOLCOL  LDA     #0      START WITH ZERO
  FBDF 85 63                 STA     LOGCOL
  FBE1 A5 54                 LDA     ROWCRS
  FBE3 85 51                 STA     HOLD1
  FBE5 A5 51         DOLCO1  LDA     HOLD1   ADD IN ROW COMPONENT
  FBE7 20 22 FC              JSR     LO1GET
  FBEA B0 0C                 BCS     DOLCO2  FOUND BEGINNING OF LINE
  FBEC A5 63                 LDA     LOGCOL  ADD 40 AND LOOK BACK ONE
  FBEE 18                    CLC
  FBEF 69 28                 ADC     #40
  FBF1 85 63                 STA     LOGCOL
  FBF3 C6 51                 DEC     HOLD1   UP ONE LINE
  FBF5 4C E5 FB              JMP     DOLCO1
  FBF8 18            DOLCO2  CLC             ADD IN COLCRS
  FBF9 A5 63                 LDA     LOGCOL
  FBFB 65 55                 ADC     COLCRS
  FBFD 85 63                 STA     LOGCOL
  FBFF 60                    RTS
                     ;
                     ;
                     ; DOBUFC: COMPUTE BUFFER COUNT AS THE NUMBER OF BYTES FROM
                     ;         BUFSTR TO END OF LOGICAL LINE WITH TRAILING SPACES REMOVED
                     ;
  FC00 20 9D FC      DOBUFC  JSR     PHACRS
```

```
                                                  LINE WITH TRAILING SPACES REMOVED
      FC00 20 90 FC      DOBUF  JSR   PHACRS

DISPLAY HANDLER -- 10-30-78 -- DISPLC

      FC03 A5 63                LDA   LOGCOL
      FC05 48                   PHA
      FC06 A5 6C                LDA   BUFSTR  START
      FC08 85 54                STA   ROWCRS
      FC0A A5 6D                LDA   BUFSTR+1
      FC0C 85 55                STA   COLCRS
      FC0E A9 01                LDA   #1
      FC10 85 6B                STA   BUFCNT
      FC12 A2 17         DOBUF1 LDX   #23     NORMAL
      FC14 A5 7B                LDA   SWPFLG  IF SWAPPED, ROW 3 IS THE LAST LINE ON SCREEN
      FC16 10 02                BPL   DOB1
      FC18 A2 03                LDX   #3
      FC1A E4 54         DOB1   CPX   ROWCRS  TEST IF CRSR IS AT LAST SCREEN POSITION
      FC1C D0 0B                BNE   DOBU1A
      FC1E A5 55                LDA   COLCRS
      FC20 C5 53                CMP   RMARGN
      FC22 D0 05                BNE   DOBU1A
      FC24 E6 6B                INC   BUFCNT  YES SO FAKE INCRSR TO AVOID SCROLLING
      FC26 4C 39 FC             JMP   DOBUF2
      FC29 20 D4 F9      DOBU1A JSR   INCRSB
      FC2C E6 6B                INC   BUFCNT
      FC2E A5 63                LDA   LOGCOL
      FC30 C5 52                CMP   LMARGN
      FC32 D0 DE                BNE   DOBUF1  NOT YET EOL
      FC34 C6 54                DEC   ROWCRS  BACK UP ONE INCRSR
      FC36 20 99 F7             JSR   CRSRLF
      FC39 20 A2 F5      DOBUF2 JSR   GETPLT  TEST CURRENT COLUMN FOR NON-ZERO DATA
      FC3C D0 17                BNE   DOBUF4  QUIT IF NON-ZERO
      FC3E C6 6B                DEC   BUFCNT  DECREMENT COUNTER
      FC40 A5 63                LDA   LOGCOL  BEGINNING OF LOGICAL LINE YET?
      FC42 C5 52                CMP   LMARGN
      FC44 F0 0F                BEQ   DOBUF4  YES, SO QUIT
      FC46 20 99 F7             JSR   CRSRLF  BACK UP CURSOR
      FC49 A5 55                LDA   COLCRS  IF LOGCOL=RMARGN, GO UP 1 ROW
      FC4B C5 53                CMP   RMARGN
      FC4D D0 02                BNE   DOBUF3
      FC4F C6 54                DEC   ROWCRS
      FC51 A5 6B         DOBUF3 LDA   BUFCNT
      FC53 D0 E4                BNE   DOBUF2  LOOP UNLESS BUFCNT JUST WENT TO ZERO
      FC55 68            DOBUF4 PLA
      FC56 85 63                STA   LOGCOL
      FC58 20 A8 FC             JSR   PLACRS
      FC5B 60                   RTS


                         STRBEG  MOVE BUFSTR TO BEGINNING OF LOGICAL LINE

      FC5C 20 DD FB      STRBEG JSR   DOLCOL  USE DOLCOL TO POINT HOLD1 AT BOL
      FC5F A5 51                LDA   HOLD1
      FC61 85 6C                STA   BUFSTR
      FC63 A5 52                LDA   LMARGN
      FC65 85 6D                STA   BUFSTR+1
      FC67 60                   RTS
```

```
                        ;
                        ; DELTIM  TIME TO DELETE A LINE IF IT IS EMPTY AND AN EXTENSION
                        ;
FC68 A5 63              DELTIA  LDA     LOGCOL  IF LOGCOL<>LMARGN
FC6A C5 52                      CMP     LMARGN  THEN DONT MOVE UP ONE
FC6C D0 02                      BNE     DELTIB  LINE BEFORE TESTING DELTIM
FC6E C6 54                      DEC     ROWCRS
FC70 20 DD F8           DELTIB  JSR     COLCOL
FC73 A5 63              DELTIM  LDA     LOGCOL  TEST FOR EXTENSION
FC75 C5 52                      CMP     LMARGN
FC77 F0 13                      BEQ     DELTI3  NO
FC79 20 47 F9                   JSR     CONVRT
FC7C A5 53                      LDA     RMARGN  SET UP COUNT
FC7E 38                         SEC
FC7F E5 52                      SBC     LMARGN
FC81 A8                         TAY
FC82 B1 64              DELTI1  LDA     (ADRESS),Y
FC84 D0 06                      BNE     DELTI3  FOUND A NON-0 SO QUIT AND RETURN
FC86 88                         DEY
FC87 10 F9                      BPL     DELTI1
FC89 4C DB F8           DELTI2  JMP     DELLIB  DELETE A LINE AND RETURN
FC8C 60                 DELTI3  RTS
                        ;
                        ;
                        ; TSTCTL  SEARCH CNTRLS TABLE TO SEE IF ATACHR IS A CNTL CHAR
                        ;
FC8D A2 2D              TSTCTL  LDX     #45     PREPARE TO SEARCH TABLE
FC8F BD C6 FE           TSTCT1  LDA     CNTRLS,X
FC92 CD FB 02                   CMP     ATACHR
FC95 F0 05                      BEQ     TSTCT2
FC97 CA                         DEX
FC98 CA                         DEX
FC99 CA                         DEX
FC9A 10 F3                      BPL     TSTCT1
FC9C 60                 TSTCT2  RTS
                        ;
                        ;
                        ;
                        ; PUSH ROWCRS,COLCRS AND COLCRS+1
                        ;
FC9D A2 02              PHACRS  LDX     #2
FC9F B5 54              PHACR1  LDA     ROWCRS,X
FCA1 9D B8 02                   STA     TMPROW,X
FCA4 CA                         DEX
FCA5 10 F8                      BPL     PHACR1
FCA7 60                         RTS
                        ;
                        ;
                        ; PULL COLCRS+1,COLCRS AND ROWCRS
                        ;
FCA8 A2 02              PLACRS  LDX     #2
FCAA BD B8 02           PLACR1  LDA     TMPROW,X
FCAD 95 54                      STA     ROWCRS,X
FCAF CA                         DEX
FCB0 10 F8                      BPL     PLACR1
FCB2 60                         RTS
                        ;
                        ;
                        ;
                        ; SWAP  IF MIXED MODE, SWAP TEXT CURSORS WITH REGULAR CURSORS
```

```
                           ; SWAP:  IF MIXED MODE, SWAP TEXT CURSORS WITH REGULAR CURSORS
```

```
       FCB3 20 B9 FC        SWAP     JSR     SWAPA   ;THIS ENTRY POINT DOES RETURN
       FCB6 4C 34 F6                 JMP     RETUR1
       FCB9 AD BF 02        SWAPA    LDA     BOTSCR
       FCBC C9 18                    CMP     #24
       FCBE F0 17                    BEQ     SWAP3
       FCC0 A2 0B                    LDX     #11
       FCC2 B5 54           SWAP1    LDA     ROWCRS,X
       FCC4 48                       PHA
       FCC5 BD 90 02                 LDA     TXTROW,X
       FCC8 95 54                    STA     ROWCRS,X
       FCCA 68                       PLA
       FCCB 9D 90 02                 STA     TXTROW,X
       FCCE CA                       DEX
       FCCF 10 F1                    BPL     SWAP1
       FCD1 A5 7B                    LDA     SWPFLG
       FCD3 49 FF                    EOR     #$FF
       FCD5 85 7B                    STA     SWPFLG
       FCD7 60              SWAP3    RTS

                            ;
                            ; CLICK:  MAKE CLICK THROUGH KEYBOARD SPEAKER
                            ;
       FCD8 A2 7F           CLICK    LDX     #$7F
       FCDA 8E 1F D0        CLICK1   STX     CONSOL
       FCDD 8E 0A D4                 STX     WSYNC
       FCE0 CA                       DEX
       FCE1 10 F7                    BPL     CLICK1
       FCE3 60                       RTS

                            ;
                            ; COLCR:  PUTS EITHER 0 OR LMARGN INTO COLCRS BASED ON MODE AND SWPFLG
                            ;
       FCE4 A9 00           COLCR    LDA     #0
       FCE6 A6 7B                    LDX     SWPFLG
       FCE8 D0 04                    BNE     COLCR1
       FCEA A6 57                    LDX     DINDEX
       FCEC D0 02                    BNE     COLCR2
       FCEE A5 52           COLCR1   LDA     LMARGN
       FCF0 85 55           COLCR2   STA     COLCRS
       FCF2 60                       RTS

                            ;
                            ; PUTMSC:  PUT SAVMSC INTO ADRESS
                            ;
       FCF3 A5 58           PUTMSC   LDA     SAVMSC  ;SET UP ADDRESS
       FCF5 85 64                    STA     ADRESS
       FCF7 A5 59                    LDA     SAVMSC+1
       FCF9 85 65                    STA     ADRESS+1
       FCFB 60                       RTS
```

```
                                            ROWCRS,COLCRS
                      ; DRAW -- DRAW A LINE FROM OLDROW,OLDCOL TO ~~NEWROW,NEWCOL~~
                      ; (THE AL MILLER METHOD FROM BASKETBALL)
FCFC A2 00      DRAW    LDX    #0
FCFE A5 22              LDA    ICCOMZ  TEST COMMAND: $11=DRAW  $12=FILL
FD00 C9 11              CMP    #$11
FD02 F0 08              BEQ    DRAWA
FD04 C9 12              CMP    #$12     TEST FILL
FD06 F0 03              BEQ    DRAWB    YES
FD08 A0 84              LDY    #NVALID NO, SO RETURN INVALID COMMAND
FD0A 60                 RTS
FD0B E8         DRAWB   INX
FD0C 8E B7 02   DRAWA   STX    FILFLG
FD0F A5 54              LDA    ROWCRS  PUT CURSOR INTO NEWROW,NEWCOL
FD11 85 60              STA    NEWROW
FD13 A5 55              LDA    COLCRS                                     NEWROW := ROWCRS
FD15 85 61              STA    NEWCOL
FD17 A5 56              LDA    COLCRS+1
FD19 85 62              STA    NEWCOL+1                                   NEWCOL := COLCRS
FD1B A9 01              LDA    #1                                         ROWINC := COLINC := 1
FD1D 85 79              STA    ROWINC  SET UP INITIAL DIRECTIONS
FD1F 85 7A              STA    COLINC                                     DELTAR := NEWROW - OLDROW
FD21 38                 SEC
FD22 A5 60              LDA    NEWROW            DETERMINE DELTA ROW
FD24 E5 5A              SBC    OLDROW                                     IF DELTAR < 0 THEN
FD26 85 76              STA    DELTAR
FD28 B0 0D              BCS    DRAW1             DO DIRECTION AND ABSOLUTE VALUE   DELTAR := -DELTAR
FD2A A9 FF              LDA    #$FF              BORROW WAS ATTEMPTED
FD2C 85 79              STA    ROWINC            SET DIRECTION=DOWN                ROWINC := -1
FD2E A5 76              LDA    DELTAR
FD30 49 FF              EOR    #$FF              DELTAR = |DELTAR|
FD32 18                 CLC
FD33 69 01              ADC    #1
FD35 85 76              STA    DELTAR
FD37 38         DRAW1   SEC                                               DELTAC := NEWCOL - OLDCOL
FD38 A5 61              LDA    NEWCOL            NOW DELTA COLUMN
FD3A E5 5B              SBC    OLDCOL
FD3C 85 77              STA    DELTAC
FD3E A5 62              LDA    NEWCOL+1          TWO-BYTE QUANTITY
FD40 E5 5C              SBC    OLDCOL+1
FD42 85 78              STA    DELTAC+1
FD44 B0 16              BCS    DRAW2             DIRECTION AND ABSOLUTE VALUE IF DELTAC < 0 THEN
FD46 A9 FF              LDA    #$FF              BORROW WAS ATTEMPTED
FD48 85 7A              STA    COLINC            SET DIRECTION = LEFT          DELTAC := -DELTAC
FD4A A5 77              LDA    DELTAC                                          COLINC := -1
FD4C 49 FF              EOR    #$FF              DELTAC = |DELTAC|
FD4E 85 77              STA    DELTAC
FD50 A5 78              LDA    DELTAC+1
FD52 49 FF              EOR    #$FF
FD54 85 78              STA    DELTAC+1
FD56 E6 77              INC    DELTAC            ADD ONE FOR TWOS COMPLEMENT
FD58 D0 02              BNE    DRAW2
FD5A E6 78              INC    DELTAC+1
FD5C A2 02      DRAW2   LDX    #2      ZERO RAM FOR DRAW LOOP
FD5E A0 00              LDY    #0
FD60 84 73              STY    COLAC+1
FD62 98         DRAW3A  TYA
FD63 95 70              STA    ROWAC,X                                    ROWAC := 0
FD65 B5 5A              LDA    OLDROW,X                                   COLAC := 0
```

```
FD63 95 70          STA   ROWAC,X
FD65 B5 5A          LDA   OLDROW,X
```

DISPLAY HANDLER -- 10-30-78 -- DISPLC

```
FD67 95 54          STA   ROWCRS,X
FD69 CA             DEX
FD6A 10 F6          BPL   DRAW3A
FD6C A5 77          LDA   DELTAC          FIND LARGER ONE (ROW OR COL)
                    STA   COUNTR          (PREPARE COUNTR AND ENDPT)
                    STA   ENDPT
FD6E E8             INX             MAKE X 0
FD6F A8             TAY
FD70 A5 78          LDA   DELTAC+1
FD72 85 7F          STA   COUNTR+1
FD74 85 75          STA   ENDPT+1
FD76 D0 0B          BNE   DRAW3           AUTOMATICALLY LARGER IF MSD>0
FD78 A5 77          LDA   DELTAC
FD7A C5 76          CMP   DELTAR          LOW COL >LOW ROW?
FD7C B0 05          BCS   DRAW3           YES
FD7E A5 76          LDA   DELTAR
FD80 A2 02          LDX   #2
FD82 A8             TAY
FD83 98       DRAW3 TYA             PUT IN INITIAL CONDITIONS
FD84 85 7E          STA   COUNTR
FD86 85 74          STA   ENDPT
FD88 48             PHA             SAVE AC
FD89 A5 75          LDA   ENDPT+1 PUT LSB OF HIGH BYTE
FD8B 4A             LSR   A       INTO CARRY
FD8C 68             PLA           RESTORE AC
FD8D 6A             ROR   A       ROR THE 9 BIT ACUMULATOR
FD8E 95 70          STA   ROWAC,X
FD90 A5 7E   DRAW4A LDA   COUNTR          TEST ZERO
FD92 05 7F          ORA   COUNTR+1
FD94 D0 03          BNE   DRAW11  IF COUNTER IS ZERO, LEAVE DRAW
FD96 4C 42 FE       JMP   DRAW10
FD99 18      DRAW11 CLC                   ADD ROW TO ROWAC (PLOT LOOP)
FD9A A5 70          LDA   ROWAC
FD9C 65 76          ADC   DELTAR
FD9E 85 70          STA   ROWAC
FDA0 90 02          BCC   DRAW5
FDA2 E6 71          INC   ROWAC+1
FDA4 A5 71    DRAW5 LDA   ROWAC+1         COMPARE ROW TO ENDPOINT
FDA6 C5 75          CMP   ENDPT+1         IF HIGH BYTE OF ROW IS .LT. HIGH
FDA8 90 14          BCC   DRAW6           BYTE OF ENDPT, BLT TO COLUMN
FDAA D0 06          BNE   DRAW5A
FDAC A5 70          LDA   ROWAC
FDAE C5 74          CMP   ENDPT           LOW BYTE
FDB0 90 0C          BCC   DRAW6           ALSO BLT
FDB2 18      DRAW5A CLC                   GE SO MOVE POINT
FDB3 A5 54          LDA   ROWCRS
FDB5 65 79          ADC   ROWINC
FDB7 85 54          STA   ROWCRS
FDB9 A2 00          LDX   #0              AND SUBTRACT ENDPT FROM ROWAC
FDBB 20 7A FA       JSR   SUBEND
FDBE 18       DRAW6 CLC                   DO SAME FOR COLUMN (DOUBLE BYTE ADD)
FDBF A5 72          LDA   COLAC           ADD
FDC1 65 77          ADC   DELTAC
FDC3 85 72          STA   COLAC
FDC5 A5 73          LDA   COLAC+1
FDC7 65 78          ADC   DELTAC+1
FDC9 85 73          STA   COLAC+1
FDCB C5 75          CMP   ENDPT+1         COMPARE HIGH BYTE
FDCD 90 27          BCC   DRAW8
FDCF D0 0A          BNE   DRAW6A
```

```
COLAC := 0

ROWCRS := OLDROW
COLCRS := OLDCOL

COUNTR := MAX (DELTAC, DELTAR)
ENDPT := COUNTR

IF DELTAC > DELTAR THEN
   ROWAC := ENDPT / 2
  ELSE
   COLAC := ENDPT / 2

IF COUNTR = 0 THEN EXIT (DRAW)

ROWAC := ROWAC + DELTAR

IF ROWAC >= ENDPT THEN
      ROWCRS := ROWCRS + ROWINC

      ROWAC := ROWAC - ENDPT

COLAC := COLAC + DELTAC

IF COLAC >= ENDPT THEN
      COLCRS := COLCRS + COLINC
      COLAC := COLAC - ENDPT
```

```
FDD1 A5 72                LDA    COLAC          COMPARE LOW BYTE
FDD3 C5 74                CMP    ENDPT
FDD5 90 1F                BCC    DRAW8
FDD7 24 7A       DRAW6A   BIT    COLINC  + OR - ?
FDD9 10 10                BPL    DRAW6B
FDDB C6 55                DEC    COLCRS         DO DOUBLE BYTE DECREMENT
FDDD A5 55                LDA    COLCRS
FDDF C9 FF                CMP    #$FF
FDE1 D0 0E                BNE    DRAW7
FDE3 A5 56                LDA    COLCRS+1
FDE5 F0 0A                BEQ    DRAW7          DONT DEC IF ZERO
FDE7 C6 56                DEC    COLCRS+1
FDE9 10 06                BPL    DRAW7   (UNCONDITIONAL)
FDEB E6 55       DRAW6B   INC    COLCRS         DO DOUBLE BYTE INCREMENT
FDED D0 02                BNE    DRAW7
FDEF E6 56                INC    COLCRS+1
FDF1 A2 02       DRAW7    LDX    #2             AND SUBTRACT ENDPT FROM COLAC
FDF3 20 7A FA             JSR    SUBEND
FDF6 20 96 FA    DRAW8    JSR    RANGE
FDF9 20 E0 F5             JSR    OUTPLT         PLOT POINT
FDFC AD B7 02             LDA    FILFLG  TEST RIGHT FILL
FDFF F0 2F                BEQ    DRAW9
FE01 20 9D FC             JSR    PHACRS
FE04 AD FB 02             LDA    ATACHR
FE07 8D BC 02             STA    HOLD4
FE0A A5 54       DRAW8A   LDA    ROWCRS  SAVE ROW IN CASE OF CR
FE0C 48                   PHA
FE0D 20 DC F9             JSR    INCRSA  POSITION CURSOR ONE PAST DOT
FE10 68                   PLA            RESTORE ROWCRS
FE11 85 54                STA    ROWCRS
FE13 20 96 FA    DRAW8C   JSR    RANGE
FE16 20 A2 F5             JSR    GETPLT  GET DATA
FE19 D0 0C                BNE    DRAW8B  STOP IF NON-ZERO DATA IS ENCOUNTERED
FE1B AD FD 02             LDA    FILDAT  FILL DATA
FE1E 8D FB 02             STA    ATACHR
FE21 20 E0 F5             JSR    OUTPLT  DRAW IT
FE24 4C 0A FE             JMP    DRAW8A  LOOP
FE27 AD BC 02    DRAW8B   LDA    HOLD4
FE2A 8D FB 02             STA    ATACHR
FE2D 20 AB FC             JSR    PLACRS
FE30 38          DRAW9    SEC                   DO DOUBLE BYTE SUBTRACT
FE31 A5 7E                LDA    COUNTR
FE33 E9 01                SBC    #1
FE35 85 7E                STA    COUNTR
FE37 A5 7F                LDA    COUNTR+1
FE39 E9 00                SBC    #0
FE3B 85 7F                STA    COUNTR+1
FE3D 30 03                BMI    DRAW10
FE3F 4C 90 FD             JMP    DRAW4A
FE42 4C 34 F6    DRAW10   JMP    RETUR1
```

COUNTR := COUNTR - 1

```
                          ;
                          ;  TABLES
                          ;
                          ;
                          ;  MEMORY ALLOCATION  (# of blocks of 4φ bytes)
                          ;
     FE45 18 10 0A        ALOCAT   BYTE   24,16,10,10,16,28,52,100,196,196,196,196
     FE48 0A 10 1C
     FE4B 34 64 C4
     FE4E C4 C4 C4
                          ;
                          ;
                          ;  NUMBER OF DISPLAY LIST ENTRIES
                          ;
     FE51 17 17 0B        NUMDLE   BYTE   23,23,11,23,47,47,95,95,97,97,97,97     really this number x 2 (special case)
     FE54 17 2F 2F
     FE57 5F 5F 61
     FE5A 61 61 61
     FE5D 13 13 09        MXDMDE   BYTE   19,19,9,19,39,39,79,79,65,65,65,65      ;EXTENSION OF NUMDLE
     FE60 13 27 27                                                                ~~***~~ split screen configurations
     FE63 4F 4F 41
     FE66 41 41 41
                          ;
                          ;
                          ;  ANTIC CODE FROM INTERNAL MODE CONVERSION TABLE
                          ;
                          ;    INTERNAL        ANTIC CODE        DESCRIPTION
                          ;       0                2          40X2X8   CHARACTERS
                          ;       1                6          20X5X8     ""
                          ;       2                7          20X5X16    ""
                          ;       3                8          40X4X8   GRAPHICS
                          ;       4                9          80X2X4     ""
                          ;       5                A          80X4X4     ""
                          ;       6                B          160X2X2    ""
                          ;       7                D          160X4X2    ""
                          ;       8                F          320X2X1    ""
                          ;       9          SAME AS 8 BUT GTIA 'LUM' MODE
                          ;       10         SAME AS 8 BUT GTIA 'COL/LUM REGISTER' MODE
                          ;       11         SAME AS 8 BUT GTIA 'COLOR' MODE
                          ;
     FE69 02 06 07        ANCONV   BYTE   2,6,7,8,9,$A,$B,$D,$F,$F,$F,$F  ;ZEROS FOR RANGE TEST IN PAGETB
     FE6C 08 09 0A
     FE6F 0B 0D 0F
     FE72 0F 0F 0F
                          ;
                          ;
                          ;  PAGE TABLE TELLS WHICH DISPLAY LISTS ARE IN DANGER OF
                          ;  CROSSING A 256 BYTE PAGE BOUNDARY
                          ;
     FE75 00 00 00        PAGETB   BYTE   0,0,0,0,0,0,0,0,1,1,1,1
     FE78 00 00 00
     FE7B 00 01 01
     FE7E 01 01 01
                          ;
                          ;
                          ;  THIS IS THE NUMBER OF LEFT SHIFTS NEEDED TO MULTIPLY
                          ;  COLCRS BY 10,20, OR 40.  (ROWCRS*10)/(2**DHLINE)
                          ;
     FE81 02 01 01        DHLINE   BYTE   2,1,1,0,0,1,1,2,2,2,2,2
```

```
FE84 00 00 01
FE87 01 00 00
FE8A 02 02 00

                    ; COLUMN: NUMBER OF COLUMNS
                    ;
FE8D 28 14 14       COLUMN  .BYTE  40,20,20,40,80,80,160,160,64,80,80,80  ;MODE 8 IS SPECIAL CASE
FE90 28 50 50
FE93 A0 A0 40
FE96 50 50 50
                    ;
                    ;
                    ; NOROWS: NUMBER OF ROWS
                    ;
FE99 18 18 0C       NOROWS  .BYTE  24,24,12,24,48,48,96,96,192,192,192,192
FE9C 18 30 30
FE9F 60 60 C0
FEA2 C0 C0 C0
                    ;
                    ;
                    ;
                    ; DIV2TB: HOW MANY RIGHT SHIFTS FOR HCRSR FOR PARTIAL BYTE MODES
                    ;
FEA5 00 00 00       DIV2TB  .BYTE  0,0,0,2,3,2,3,2,3,1,1,1
FEA8 02 03 02
FEAB 03 02 03
FEAE 01 01 01
                    ;
                    ;
                    ; DMASKT: DISPLAY MASK TABLE
                    ;
FEB1 00 FF F0       DMASKT  .BYTE  $00,$FF,$F0,$0F
FEB4 0F
FEB5 C0 30 0C               .BYTE  $C0,$30,$0C,$03
FEB8 03
                    ;
                    ; MASKTB: BIT MASK.   (ALSO PART OF DMASKTB! DO NOT SEPARATE)
                    ;
FEB9 80 40 20       MASKTB  .BYTE  $80,$40,$20,$10,$08,$04,$02,$01
FEBC 10 08 04
FEBF 02 01
                    ;
                    ;
                    ;
FEC1 28 CA 94       COLRTB  .BYTE  $28,$CA,$94,$46,$00
FEC4 46 00
                    ;
                    ;
                    ;
                    ;
                    ; CNTRLS: CONTROL CODES AND THEIR DISPLACEMENTS INTO THE
                    ;         CONTROL CHARACTER PROCESSORS
                    ;
FEC6 1B             CNTRLS  .BYTE  $1B
FEC7 79 F7                  .WORD  ESCAPE
FEC9 1C                     .BYTE  $1C
```

```
FECA 7F F7                    WORD   CRSRUP
FECC 1D                       BYTE   $1D
FECD 8C F7                    WORD   CRSRDN
FECF 1E                       BYTE   $1E
FED0 99 F7                    WORD   CRSRLF
FED2 1F                       BYTE   $1F
FED3 AA F7                    WORD   CRSRRT
FED5 7D                       BYTE   $7D
FED6 B9 F7                    WORD   CLRSCR
FED8 7E                       BYTE   $7E
FED9 E6 F7                    WORD   BS
FEDB 7F                       BYTE   $7F
FEDC 10 F8                    WORD   TAB
FEDE 9B                       BYTE   $9B
FEDF 30 FA                    WORD   DOCRWB
FEE1 9C                       BYTE   $9C
FEE2 D4 F8                    WORD   DELLIN
FEE4 9D                       BYTE   $9D
FEE5 A4 F8                    WORD   INSLIN
FEE7 9E                       BYTE   $9E
FEE8 32 F8                    WORD   CLRTAB
FEEA 9F                       BYTE   $9F
FEEB 2D F8                    WORD   SETTAB
FEED FD                       BYTE   $FD
FEEE 0A F9                    WORD   BELL
FEF0 FE                       BYTE   $FE
FEF1 6D F8                    WORD   DELCHR
FEF3 FF                       BYTE   $FF
FEF4 37 F8                    WORD   INSCHR



                        ATAINT  ATASCI TO INTERNAL TABLE

FEF6 40 00 20           ATAINT    BYTE   $40,$00,$20,$60
FEF9 60


                        INTATA  INTERNAL TO ATASCI TABLE

FEFA 20 40 00           INTATA    BYTE   $20,$40,$00,$60
FEFD 60


                        ATASCI  ATASCII CONVERSION TABLE

FEFE 6C 6A 3B           ATASCI    BYTE   $6C,$6A,$3B,$80,$80,$6B,$2B,$2A LOWER CASE
FF01 80 80 6B
FF04 2B 2A
FF06 6F 80 70                     BYTE   $6F,$80,$70,$75,$9B,$69,$2D,$3D
FF09 75 9B 69
FF0C 2D 3D

FF0E 76 80 63                     BYTE   $76,$80,$63,$80,$80,$62,$78,$7A
FF11 80 80 62
FF14 78 7A
FF16 34 80 33                     BYTE   $34,$80,$33,$36,$1B,$35,$32,$31
FF19 36 1B 35
```

special:
80 ≡ INVALID CODE
81 ≡ JIL
82 ≡ CAPS
83 ≡ SHIFT-CAPS
84 ≡ CTRL-CAPS
85 ≡ EOF

```
FF1C 30 31

FF1E 2C 20 2E              BYTE   $2C,$20,$2E,$6E,$80,$60,$2F,$81
FF21 6E 80 60
FF24 2F 81
FF26 72 80 65              BYTE   $72,$80,$65,$79,$7F,$74,$77,$71
FF29 79 7F 74
FF2C 77 71

FF2E 39 80 30              BYTE   $39,$80,$30,$37,$7E,$38,$3C,$3E
FF31 37 7E 38
FF34 3C 3E
FF36 66 68 64              BYTE   $66,$68,$64,$80,$82,$67,$73,$61
FF39 80 82 67
FF3C 73 61

FF3E 4C 4A 3A              BYTE   $4C,$4A,$3A,$80,$80,$4B,$5C,$5E UPPER CASE
FF41 80 80 4B
FF44 5C 5E
FF46 4F 80 50              BYTE   $4F,$80,$50,$55,$9B,$49,$5F,$70
FF49 55 9B 49
FF4C 5F 70

FF4E 56 80 43              BYTE   $56,$80,$43,$80,$80,$42,$58,$5A
FF51 80 80 42
FF54 58 5A
FF56 24 80 23              BYTE   $24,$80,$23,$26,$1B,$25,$22,$21
FF59 26 1B 25
FF5C 22 21

FF5E 5B 20 5D              BYTE   $5B,$20,$5D,$4E,$80,$4D,$3F,$81
FF61 4E 80 4D
FF64 3F 81
FF66 52 80 45              BYTE   $52,$80,$45,$59,$9F,$54,$57,$51
FF69 59 9F 54
FF6C 57 51

FF6E 28 80 29              BYTE   $28,$80,$29,$27,$9C,$40,$7D,$9D
FF71 27 9C 40
FF74 7D 9D
FF76 46 48 44              BYTE   $46,$48,$44,$80,$83,$47,$53,$41
FF79 80 83 47
FF7C 53 41


FF7E 0C 0A 7B              BYTE   $0C,$0A,$7B,$80,$80,$0B,$1E,$1F CONTROL
FF81 80 80 0B
FF84 1E 1F
FF86 0F 80 10              BYTE   $0F,$80,$10,$15,$9B,$09,$1C,$1D
FF89 15 9B 09
FF8C 1C 1D

FF8E 16 80 03              BYTE   $16,$80,$03,$80,$80,$02,$18,$1A
FF91 80 80 02
FF94 18 1A
FF96 80 80 85              BYTE   $80,$80,$85,$80,$18,$80,$FD,$80
FF99 80 18 80
FF9C FD 80
```

DISPLAY HANDLER -- 10-30-78 -- DISPLC

```
  FF9E 00 20 60           BYTE   $00,$20,$60,$0E,$80,$0D,$80,$81
  FFA1 0E 80 0D
  FFA4 80 81
  FFA6 12 80 05           BYTE   $12,$80,$05,$19,$9E,$14,$17,$11
  FFA9 19 9E 14
  FFAC 17 11

  FFAE 80 80 80           BYTE   $80,$80,$80,$80,$FE,$80,$7D,$FF
  FFB1 80 FE 80
  FFB4 7D FF
  FFB6 06 08 04           BYTE   $06,$08,$04,$80,$84,$07,$13,$01
  FFB9 80 84 07
  FFBC 13 01



  FFBE AD 09 02    PIRQ   LDA    KBCODE
  FFC1 CD F2 02           CMP    CH1      TEST AGAINST LAST KEY PRESSED
  FFC4 D0 05              BNE    PIRQ3    IF NOT, GO PROCESS KEY
  FFC6 AD F1 02           LDA    KEYDEL   IF KEY DELAY BYTE > 0
  FFC9 D0 20              BNE    PIRQ4    IGNORE KEY AS BOUNCE          make reassignable
  FFCB AD 09 02    PIRQ3  LDA    KBCODE   RESTORE AC
  FFCE C9 9F              CMP    #CNTL1   TEST CONTROL 1 (SSFLAG)
  FFD0 D0 0A              BNE    PIRQ1
  FFD2 AD FF 02           LDA    SSFLAG                           doesn't go to CH1 -- no debounce
  FFD5 49 FF              EOR    #$FF
  FFD7 8D FF 02           STA    SSFLAG
  FFDA B0 0F              BCS    PIRQ4    (UNCONDITIONAL) MAKE ^1 INVISIBLE
  FFDC 8D FC 02    PIRQ1  STA    CH
  FFDF 8D F2 02           STA    CH1
  FFE2 A9 03              LDA    #3
  FFE4 8D F1 02           STA    KEYDEL   INITIALIZE KEY DELAY FOR DEBOUNCE
  FFE7 A9 00              LDA    #0       CLEAR COLOR-SHIFT BYTE
  FFE9 85 4D              STA    ATRACT
  FFEB A9 30       PIRQ4  LDA    #$30     3/4 SECOND DELAY FOR AUTO-REPEAT.
  FFED 8D 2B 02           STA    SRTIMR
  FFF0 68          PIRQ2  PLA
  FFF1 40                 RTI

  FFF2 FF FF FF           BYTE   $FF,$FF,$FF,$FF,$FF,$FF
  FFF5 FF FF FF

  FFFB             CRNTPC =*
                          *=$14
  0014 00          KBDSPR BYTE   $FFFB-CRNTPC    DISPLC IS TOO LONG
                          END
```

```
DISPLAY HANDLER -- 10-30-79 -- DISPLC

           SYMBOL  TABLE
[illegible] 0006   ADRESS  0064    AFP     D800    ALLPOT  020D
[illegible] FE45   ANCONV  [illegible]  ANTIC   D400    APPEND  0001
[illegible] 000E   ASCCOL  F705    ATACHR  02FB    ATAINT  FEF6
[illegible] BE43   ATASCI  FEFE    ATRACT  004D    AUDC1   D201
AUDC2   D203    AUDC3   D205    AUDC4   D207    AUDCTL  D208
AUDF1   D200    AUDF2   D202    AUDF3   D204    AUDF4   D206
[illegible] 0086   BADMOD  0091    BELL    F90A    BELL1   F90C
BFENHI  0035    BFENLO  0034    BITCLR  FB12    BITCON  FAE8
[illegible] FB31   BITGET  FB25    BITMSK  006E    BITPUT  FB04
[illegible] FAFA   BITSET  FB06    BLIM    028A    BLKBDV  E471
BOOT?   0009    BOOTAD  0242    BOTSCR  02BF    BPTR    003D
BRKABT  0080    BRKKEY  0011    BS      F7E6    BS1     FB0D
BS2     F805    BS3     F7F5    BSA     F7EC    BUFADR  0015
BUFCNT  006B    BUFRFL  0038    BUFRHI  0033    BUFRLO  0032
BUFSTR  006C    CASBUF  03FD    CASETV  E440    CASFLG  030F
CASINI  0002    CASORG  EF41    CASSBT  004B    CASSET  0043
CAUX1   023C    CAUX2   023D    CBAUDH  02EF    CBAUDL  02EE
CCOMND  023B    CDEVIC  023A    CDTMA1  0226    CDTMA2  0228
CDTMF3  022A    CDTMF4  022C    CDTMF5  022E    CDTMV1  0218
CDTMV2  021A    CDTMV3  021C    CDTMV4  021E    CDTMV5  0220
CH      02FC    CH1     02F2    CHACT   02F3    CHACTL  D401
CHAR    02FA    CHBAS   02F4    CHBASE  D409    CHKERR  008F
CHKSNT  003B    CHKSUM  0031    CHRORG  E000    CIOCHR  002F
CIOINV  E46E    CIOORG  E4A6    CIOV    E456    CIX     00F2
CKEY    004A    CLICK   FCD8    CLICK1  FCDA    CLOSE   000C
CLRCOD  007D    CLRLI1  FBA6    CLRLIN  FB9B    CLRSC2  F7BF
CLRSC3  F7CE    CLRSCR  F7D9    CLRTAB  F832    CLRTBS  F430
[illegible] 009F   CNTRLS  FEC6    COLAC   0072    COLBK   D01A
[illegible] FCE4   COLCR1  FCBE    COLCR2  FCF0    COLCRS  0055
COLDST  0244    COLDSV  E477    COLINC  007A    COLOR0  02C4
COLOR1  02C5    COLOR2  02C6    COLOR3  02C7    COLOR4  02C8
COLPF0  D016    COLPF1  D017    COLPF2  D018    COLPF3  D019
COLPM0  D012    COLPM1  D013    COLPM2  D014    COLPM3  D015
COLRSH  004F    COLRTB  FEC1    COLUMN  FE8D    COMRE1  F7A7
COMRET  F789    CONSOL  D01F    CONVR1  F97E    CONVR2  F98B
CONVR3  F98F    CONVR4  F99C    CONVR5  F9A6    CONVR6  F9A7
CONVR7  F947    COS     0073    COUNTR  007E    CR      009B
CRETRY  0036    CRITIC  0042    CRLOOP  F5E7    CRNTPC  FFF9
CRSINH  02F0    CRSRDN  F78C    CRSRL1  F7A3    CRSRLF  F799
[illegible] 008D   CRSRRT  F7AA    CRSRUP  F77F    CSOPIV  E47D
CSTAT   0288    CTIA    D000    DAUX1   030A    DAUX2   030B
[illegible] F913   DBDEC   F91F    DBSECT  0241    DBSWB   F921
[illegible] F934   DBUFHI  0305    DBUFLO  0304    DBYTHI  0309
DBYTLO  0308    DCB     0300    DCOMND  0302    DDEVIC  0300
DEGFLG  00FB    DEGON   0006    DELCH1  F870    DELCH2  F896
DELCHR  F86D    DELETE  0051    DELLI1  F8DD    DELLI2  F8F8
DELLIA  F807    DELLIB  F8D0    DELLIN  F8D4    DELTAC  0077
DELTAR  0076    DELTI1  FC83    DELTI2  FC89    DELTI3  FC8C
DELTIA  FC6B    DELTIB  FC70    DELTIM  FC73    DERROR  0090
DFLAGS  0240    DHLINE  FEB1    DIORT   00F1    DINDEX  0057
DIRECT  0002    DISK    0044    DISKIV  E450    DISPLA  E410
DISPLY  0053    DIV2TB  FEA5    DLISTH  D403    DLISTL  D402
DMACTL  D400    DMASK   02A0    DMASKT  FEB1    DNACK   006B
[illegible] FC1A   DOBU1A  FC29    DOBUF1  FC12    DOBUF2  FC39
DOBUF3  FC51    DOBUF4  FC55    DOBUFC  FC00    DOCR-   FA34
DOCR1   FA00    DOCR1A  FA25    DOCR1B  FA14    DOCR2   FA3D
DOCR2A  FA4A    DOCR2B  FA4D    DOCR4B  FA61    DOCRWS  FA30
DOLCO1  FBE5    DOLCO2  FBF8    DOLCOL  FB00    DOPEN   F3F6
DOPEN1  F460    DOPEN2  F4D5    DOPEN3  F4FA    DOPEN4  F51C
```

```
DOPEN1  F4B0     DOPEN2  F4D5     DOPEN3  F4FA     DOPEN4  F51C

DISPLAY HANDLER -- 10-30-78 -- DISPLC

DOPEN6  F529     DOPEN7  F588     DOPEN8  F438     DOPEN9  F577
DOPENA  F457     DOSINT  000C     DOSS    F6AD     DOSVEC  000A
DRAW    FCFC     DRAW1   FD37     DRAW10  FE42     DRAW11  FD99
DRAW2   FD9C     DRAW3   FD37     DRAW3A  FD62     DRAW4A  FD90
DRAW5   FDA4     DRAW5A  FDB1     DRAW6   FDBE     DRAW6A  FDD7
DRAW6B  FDEB     DRAW7   FDF1     DRAW8   FDF6     DRAW8A  FE0A
DRAW8B  FE27     DRAW8C  FE13     DRAW9   FE30     DRAWA   FD0C
DRAWB   FD0B     DRAWLN  0011     DRETRY  0037     DRKMSK  004E
DSKFMS  0018     DSKINV  E453     DSKORG  EDEA     DSKTIM  0246
DSKUTL  001A     DSPFLG  02FE     DSTAT   004C     DSTATS  0303
DTIMLO  0306     DUNIT   0301     DUNUSE  0307     DVSTAT  02EA
EDITOR  E400     EDITRV  E400     EEXP    00ED     EGETC1  F650
EGETC2  F65E     EGETC3  F67C     EGETC5  F691     EGETC6  F6A0
EGETC7  F6B0     EGETCH  F63E     ENDPT   0074     EOFERR  0088
EOPEN   F3FC     EOUTC5  F6BE     EOUTC6  F6B5     EOUTCH  F6A4
ERANGE  FA89     ERETN   F6BB     ERRFLG  023F     ESCAPE  F779
ESCFLG  02A2     ESIGN   00EF     EXP     DDC0     EXP10   DDCC
EXTEN2  F87E     EXTEN3  F886     EXTEN4  F892     EXTEND  F878
FADD    DA66     FASC    D8E6     FCHRFL  00F0     FDIV    DB28
FEOF    003F     FILDAT  02FD     FILFLG  02B7     FILLIN  0012
FLD0P   DD8D     FLD0R   DD89     FLD1P   DD9C     FLD1R   DD98
FLPTR   00FC     FMOVE   DDB6     FMSZPG  0043     FMUL    DADB
[illegible]  0092     FORMAT  0022     FPI     D9D2     FPREC   0036
FPSCR   05E6     FPSCR1  05EC     FPTR2   00FE     FR0     00D4
FR1     00E0     FR2     00E6     FRE     00DA     FREQ    0040
FRMADR  0068     FRMERR  008C     FRX     00EC     FSCR    05E6
FSCR1   05EC     FSTOP   DDAD     FSTOR   DDA7     FSUB    DA60
FTYPE   003E     GETCH   F593     GETCHR  0007     GETOUT  F749
GETPLT  F5A2     GETREC  0005     GLBABS  02E0     GPRIOR  026F
GRACTL  D01D     GRAFM   D011     GRAFP0  D00D     GRAFP1  D00E
GRAFP2  D00F     GRAFP3  D010     HATABS  031A     HITCLR  D01E
HOLD1   0051     HOLD2   029F     HOLD3   029D     HOLD4   02BC
HOLD5   02BD     HOLDCH  007C     HOME    F7D6     HPOSM0  D004
HPOSM1  D005     HPOSM2  D006     HPOSM3  D007     HPOSP0  D000
HPOSP1  D001     HPOSP2  D002     HPOSP3  D003     HSCROL  D404
ICAX1   034A     ICAX1Z  002A     ICAX2   034B     ICAX2Z  002B
ICBAH   0345     ICBAHZ  0025     ICBAL   0344     ICBALZ  0024
ICBLH   0349     ICBLHZ  0029     ICBLL   0348     ICBLLZ  0028
ICCOM   0342     ICCOMT  0017     ICCOMZ  0022     ICDNO   0341
ICDNOZ  0021     ICHID   0340     ICHIDZ  0020     ICIDNO  002E
ICPTH   0347     ICPTHZ  0027     ICPTL   0346     ICPTLZ  0026
ICSPR   034C     ICSPRZ  002C     ICSTA   0343     ICSTAZ  0023
IFP     D9AA     INATA1  F84A     INATAC  F892     INBUFF  00F3
INCRA   F9F8     INCRS1  FA77     INCRS2  F9E4     INCRS9  F9F7
INCRSA  F9DC     INCRSB  F9D4     INCRSC  F9DA     INCRSR  F9D8
INSCH1  F852     INSCH3  F85E     INSCH4  F844     INSCH5  F86A
INSCH6  F861     INSCHR  F83?     INSCLR  0020     INSDAT  007D
INSLI1  F8C6     INSLI2  F8CE     INSLIA  F8A5     INSLIN  F8A6
INTABS  0200     INTATA  FEFA     INTEMP  022D     INTINV  EAA8
INTORS  E6D9     INTRPS  0010     INVFLG  02B6     IOCB    0340
IOCBAS  0020     IOCBSZ  0010     IOCFRE  00FF     IRQEN   D20E
IRQST   D20E     JSRIND  F6A1     K1      F729     K2      F734
K3      F73F     K4      F726     K5      F768     K6      F740
K7      F745     K8      F773     KBCODE  D209     KBD     004B
KBDHND  E420     KBDORG  F3E4     KBDSPR  0014     KEYBDV  E420
KEYDEL  02F1     KGETC1  F71E     KGETC2  F6DD     KGETC3  F6F5
KGETCH  F6E2     LBFEND  05FF     LBPR1   057E     LBPR2   057F
LBUFF   0580     LEDGE   0002     LFRTCH  F7A5     LINBUF  0247
LMARGN  0088     LNARPM  0002     LOGGET  F822     LOGRET  F829
[illegible]  0023     LOC     DE10     LOG10   DED1     LOGCOL  0063
```

```
DISPLAY HANDLER — 10-30-78 — DISPLC

[illegible] F620    LOGMAP  02B2    LPENH   0234    LPENV   0235
M0PF    D000    M0PL    D008    M1PF    D001    M1PL    D009
M2PF    D002    M2PL    D00A    M3PF    D003    M3PL    D00B
MACHID  FEB9    MAXDEV  0021    MAXIOC  0080    MEMLO   02E7
MEMTOP  02E5    MLTTMP  0066    MODEH   004D    MONORG  F063
MOVLI1  F85B    MOVLI2  F87A    MOVLIN  F84E    MXDMDE  FE50
[illegible] 0010    NEWCOL  0061    NEWROW  0060    NMIEN   D40E
NMIRES  D40F    NMIST   D40F    NOCKSM  003C    NOFUNC  F630
NOMOD   F4AB    NONDEV  0082    NOROWS  FE99    NOSCR1  FA32
[illegible] FA2C    NOTB    F48B    NOTE    0026    NOTMXD  F4F5
[illegible] 0086    NSIGN   00EE    NUMDLE  FE51    NVALID  0084
[illegible] FAE4    OLDADR  005E    OLDCHR  005D    OLDCOL  005B
OLDROW  005A    OPEN    0003    OPNCOM  F404    OPNERR  F453
OPNIN   0004    OPNINO  000C    OPNOT   0008    OPNTMP  0066
OUTCH   F5B7    OUTCH2  F5FF    OUTCHA  F5BD    OUTCHB  F5D7
OUTCHE  F5CA    OUTPLT  F5E0    OVRRUN  008E    P0PF    D004
P0PL    D00C    P1PF    D005    P1PL    D00D    P2PF    D006
P2PL    D00E    P3PF    D007    P3PL    D00F    PACTL   D302
PADDL0  0270    PADDL1  0271    PADDL2  0272    PADDL3  0273
PADDL4  0274    PADDL5  0275    PADDL6  0276    PADDL7  0277
PAGETB  FE75    PBCTL   D303    PBPNT   001D    PBUFSZ  001E
PCOLR0  02C0    PCOLR1  02C1    PCOLR2  02C2    PCOLR3  02C3
PENH    D40C    PENV    D40D    PHACR1  FC9F    PHACRS  FC9D
PIA     D300    PIRQ0   FFBE    PIRQ1   FFDC    PIRQ2   FFF0
PIRQ3   FFCB    PIRQ4   FFEB    PLACR1  FCAA    PLACRS  FCA8
PLYARG  05E0    PLYEVL  0D40    PMBASE  D407    POINT   0025
POKEY   D200    POKMSK  0010    PORTA   D300    PORTB   D301
POT0    D200    POT1    D201    POT2    D202    POT3    D203
POT4    D204    POT5    D205    POT6    D206    POT7    D207
POTGO   D20B    PRINTR  0050    PRINTV  E430    PRIOR   D01B
PRNBUF  03C0    PRNORG  EE78    PRVOPN  0081    PTEMP   001F
PTIMOT  001C    PTRIG0  027C    PTRIG1  027D    PTRIG2  027E
PTRIG3  027F    PTRIG4  0280    PTRIG5  0281    PTRIG6  0282
PTRIG7  0283    PUTCHR  000B    PUTMSC  FCF3    PUTREC  0009
PWRONA  F3E4    RADFLG  00FB    RADON   0000    RAMLO   0004
RAMSIZ  02E4    RAMTOP  006A    RANDOM  D20A    RANGE   FA96
RANGE1  FAB7    RANGE2  FABE    RANGEZ  FA9E    RBLOKV  E47A
RDONLY  0087    RECVDN  0039    REDGE   0027    RENAME  0020
RETUR1  F634    RETUR3  FAE1    RETURN  F621    RMARGN  0053
RNGER1  FAD8    RNGER2  FAD6    RNGERR  FAD1    RNGOK   FAC4
ROWAC   0070    ROWCRS  0054    ROWINC  0079    RTCLOK  0012
SAVADR  0068    SAVIO   0316    SAVMSC  0058    SCREDT  0045
SCRENV  E410    SCRFLG  02BB    SCRMEM  0093    SCROL1  F8B7
SCROL2  F8CA    SCROLL  F8AC    SDLSTH  0231    SDLSTL  0230
SDMCTL  022F    SENDEV  E468    SERIN   D20D    SEROUT  D20D
SETTAB  F82D    SETVBV  E45C    SHFAMT  006F    SHFLOK  02BE
SHIFT1  F5B1    SHIFT2  F610    SHIFTD  F5AA    SHIFTU  F608
[illegible] E0B1    SIOINV  E465    SIOORG  E944    SIOV    E459
SIZEM   D00C    SIZEP0  D008    SIZEP1  D009    SIZEP2  D00A
SIZEP3  D00B    SKCTL   D20F    SKRES   D20A    SKSTAT  D20F
SOUNDR  0041    SPECIL  000E    SQR     DE81    SRTIMR  022B
SSFLAG  02FF    SSKCTL  0232    STACKP  0318    STATIS  0000
STATUS  0030    STICK0  0278    STICK1  0279    STICK2  027A
STICK3  027B    STIMER  D209    STORE   F917    STORE1  F910
[illegible] F05C    STRERR  F442    STRIG0  0284    STRIG1  0285
STRIG2  0286    STRIG3  0287    [illegible] F94E    SUBEND  FA7A
SUBTMP  029E    SUCCES  0001    SWAP    FC83    SWAP1   FCC2
SWAP2   FCD7    SWAPA   FC19    SWPFLG  007B    SYSVBV  E45F
TAB     F810    TAB1    F823    TAB2    F824    TABMAP  02A3
TEMP    023E    TEMP1   0313    TEMP2   0314    TEMP3   0315
```

```
DISPLAY HANDLER -- 10-30-78 -- DISPLC

TIMER1  030C    TIMER2  0310    TIMFLG  0317    TIMOUT  000A
TINDEX  0293    TMPCHR  0050    TMPCOL  0289    TMPLBT  02A1
TMPROW  0288    TMPX1   029C    TOADR   0066    TRAMSZ  0006
TRIG0   D010    TRIG1   D011    TRIG2   D012    TRIG3   D013
TRMNCR  0089    TSTAT   0019    TSTCT1  FC8F    TSTCT2  FC9C
TSTCTL  FC8D    TSTDAT  0007    TXTCOL  0291    TXTMSC  0294
TXTOLD  0296    TXTROW  0290    UNLOCK  D024    UPDNCM  F787
USAREA  0480    VBREAK  0206    VBWAIT  F496    VCOUNT  D40B
VCTABL  E480    VDELAY  D01C    VDSLST  0200    VECTBL  E400
VIMIRQ  0216    VINTER  0204    VKEYBD  0208    VPRCED  0202
VSCROL  D405    VSERIN  020A    VSEROC  020E    VSEROR  020C
VTIMR1  0210    VTIMR2  0212    VTIMR4  0214    VVBLKD  0224
VVBLKI  0222    WARMST  0008    WARMSV  E474    WMODE   0289
WRONLY  0083    WSYNC   D40A    XITVBV  E462    XMTDON  003A
ZIOCB   0020    ZTEMP1  00F5    ZTEMP3  00F9    ZTEMP4  00F7
```

```
 8104      C000                 PAGE    'FLOATING POINT'
 8105                           ORG     FPORG
 8106                           LOCAL
 8107                 *
 8108                 *         ASCIN - CONVERT ASCII INPUT TO INTERNAL FORM
 8109                 *
 8110                 *         ON ENTRY   INBUFF - POINTS TO BUFFER WITH ASCII
 8111                 *                    CIX - INDEX TO 1ST BYTE OF #
 8112                 *
 8113                 *         ON EXIT    CC SET - CARRY SET IF NOT #
 8114                 *                             CARRY CLEAR OF #
 8115                 *
 8116                 AFP
 8117                 CVAFP
 8118                 ASCIN
 8119 D800 2041DB               JSR     SKPBLANK
 8120 D803 2080DB               JSR     !TSTCHAR       ; SEE IF THIS COULD BE A NUMBER     CIX
 8121 D806 B039                 BCS     !NONUM         ; BR IF NOT A NUMBER
 8122                 ;
 8123                 ;         SET INITIAL VALUES
 8124                 ;
 8125 D808 A2ED                 LDX     #EEXP          ; ZERO 4 VALUES --EEXP, NSIGN, ESIGN, FCHRFLG.
 8126 D80A A004                 LDY     #4             ; X
 8127 D80C 2048DA               JSR     ZXLY           ; X
 8128 D80F A2FF                 LDX     #$FF
 8129 D811 86F1                 STX     DIGRT          ; SET TO $FF
 8130                 ;
 8131 D813 2044DA               JSR     ZFRO           ; CLEAR FRO
 8132                 ;
 8133 D816 F004                 BEQ     !IN2           ; UNCONDITIONAL BR
 8134                 ;
 8135                 ;
 8136                 !IN1
 8137 D818 A9FF                 LDA     #$FF           ; SET 1ST CHAR FLAG TO NON ZERO
 8138 D81A 85F0                 STA     FCHRFLG        ; X
 8139                 ;
 8140                 !IN2
 8141 D81C 2094DB               JSR     !GETCHAR       ; GET INPUT CHAR                CIX
 8142 D81F B021                 BCS     !NON1          ; BR IF CHAR NOT NUMBER
 8143                 ;
 8144                 ;
 8145                 ;         IT'S A NUMBER
 8146                 ;
 8147 D821 48                   PHA                    ; SAVE ON CPU STACK
 8148 D822 A6D5                 LDX     FROM           ; GET 1ST BYTE
 8149 D824 D011                 BNE     !INCE          ; INCR EXPONENT
 8150                 ;
 8151 D826 20EBDB               JSR     NIBSHO         ; SHIFT FRO ONE NIBBLE LEFT
 8152                 ;
 8153 D829 68                   PLA                    ; GET DIGIT ON CPU STACK
 8154 D82A 05D9                 ORA     FROM+FMPREC-1  ; OR INTO LAST BYTE
 8155 D82C 85D9                 STA     FROM+FMPREC-1  ; SAVE AS LAST BYTE
 8156                 ;
 8157                 ;         COUNT CHARACTERS AFTER DECIMAL POINT
 8158                  ;
 8159 D82E A6F1                  LDX     DIGRT             ; GET # OF DIGITS RIGHT
 8160 D830 30E6                  BMI     !IN1              ; IF = $FF, NO DECIMAL POINT
 8161 D832 E8                    INX                       ; ADD IN THIS CHAR
 8162 D833 86F1                  STX     DIGRT             ; SAVE
 8163 D835 D0E1                  BNE     !IN1              ; GET NEXT CHAR
 8164                  ;
 8165                  ;
 8166                  ;         INCREMENT # OR DIGIT MORE THAN 9
 8167                  ;
 8168                  ;
 8169                  !INCE
 8170 D837 68                    PLA                       ; CLEAR CPU STACK
 8171 D838 A6F1                  LDX     DIGRT             ; HAVE DP?
 8172 D83A 1002                  BPL     !INCE2            ; IF YES, DON'T INCR E COUNT
 8173 D83C E6ED                  INC     EEXP              ; INCR EXPONENT
 8174                  !INCE2
 8175 D83E 4C18D8                JMP     !IN1              ; GET NEXT CHAR
 8176                  ;
 8177                  ;
 8178                  !NONUM
 8179 D841 60                    RTS                       ; RETURN FAIL
 8180
```

```
                                  PAGE
                      ;
                      ;           NON-NUMERIC IN NUMBER BODY
                      ;
                      !NON1
D842 C92E                         CMP     #'.'              ; IS IT DECIMAL POINT?
D844 F014                         BEQ     !DP               ; IF YES, PROCESS IT
D846 C945                         CMP     #'E'              ; IS IT E FOR EXPONENT?
D848 F019                         BEQ     !EXP              ; IF YES, DO EXPONENT
                      ;
D84A A6F0                         LDX     FCHRFLG           ; IS THIS THE 1ST CHAR
D84C D06B                         BNE     !EXIT             ; IF NOT, END OF NUMERIC INPUT
D84E C92B                         CMP     #'+'              ; IS IT PLUS?
D850 F0C6                         BEQ     !IN1              ; GO FOR NEXT CHAR
D852 C92D                         CMP     #'-'              ; IS IT MINUS?
D854 F000                         BEQ     !MINUS
                      ;
     0000                         .IF     DBVER
                                  BRK                       ; SHOULDN'T GET HERE
                                  .ENDIF
                      ;
                      !MINUS
D856 85EE                         STA     NSIGN             ; SAVE SIGN FOR LATER
D858 F0BE                         BEQ     !IN1              ; UNCONDITIONAL BRANCH FOR NEXT CHAR
                      ;
                      !DP
D85A A6F1                         LDX     DIGRT             ; IS DIGRT STILL = FF?
D85C 1058                         BPL     !EXIT             ; IF NOT, ALREADY HAVE DP
D85E E8                           INX                       ; INCR TO ZERO
D85F 86F1                         STX     DIGRT             ; SAVE
D861 F0B5                         BEQ     !IN1              ; UNCONDITIONAL BR FOR NEXT CHAR
                      ;
                      !EXP
D863 A5F2                         LDA     CIX               ; GET INDEX
D865 85EC                         STA     FRX               ; SAVE
D867 2094DB                       JSR     !GETCHAR          ; GET NEXT CHAR
D86A B037                         BCS     !NON2             ; BR IF NOT NUMBER
                      ;
                      ;           IT'S A NUMBER IN AN EXPONENT
                      ;
                      !EXP2
D86C AA                           TAX                       ; SAVE 1ST CHAR OF EXPONENT
D86D A5ED                         LDA     EEXP              ; GET # OF CHAR OVER 9
D86F 48                           PHA                       ; SAVE IT
D870 86ED                         STX     EEXP              ; SAVE 1ST CHAR OF EXPONENT
D872 2094DB                       JSR     !GETCHAR          ; GET NEXT CHAR
                      ;
                      ;
D875 B017                         BCS     !EXP3             ; IF NOT # NO SECOND DIGIT
D877 48                           PHA                       ; SAVE SECOND DIGIT
                      ;
D878 A5ED                         LDA     EEXP              ; GET 1ST DIGIT
D87A 0A                           ASLA                      ; GET DIGIT * 10
D87B 85ED                         STA     EEXP              ; X
D87D 0A                           ASLA                      ; X
```

```
 8235 D87E 0A                  ASLA                          ; X
 8236 D87F 65ED                ADC     EEXP                  ; X
 8237 D881 85ED                STA     EEXP                  ; SAVE
 8238 D883 68                  PLA                           ; GET SECOND DIGIT
 8239 D884 18                  CLC
 8240 D885 65ED                ADC     EEXP                  ; GET EXPONENT INPUTTED
 8241 D887 85ED                STA     EEXP                  ; SAVE
 8242                ;
 8243 D889 A4F2                LDY     CIX                   ; INC TO NEXT CHAR
 8244 D88B 209DDB              JSR     !GCHR1                ; X
 8245                ;
 8246                ;
 8247                !EXP3
 8248 D88E A5EF                LDA     ESIGN                 ; GET SIGN OF EXPONENT
 8249 D890 F009                BEQ     !EXP1                 ; IF NO SIGN, IT IS +
 8250 D892 A5ED                LDA     EEXP                  ; GET EXPONENT ENTERED
 8251 D894 49FF                EOR     #$FF                  ; COMPLEMENT TO MAKE MINUS
 8252 D896 18                  CLC                           ; X
 8253 D897 6901                ADC     #1                    ; X
 8254 D899 85ED                STA     EEXP                  ; SAVE
 8255                !EXP1
 8256 D89B 68                  PLA                           ; GET # DIGITS MORE THAN 9
 8257 D89C 18                  CLC                           ; CLEAR CARRY
 8258 D89D 65ED                ADC     EEXP                  ; ADD IN ENTERED EXPONENT
 8259 D89F 85ED                STA     EEXP                  ; SAVE EXPONENT
 8260 D8A1 D013                BNE     !EXIT                 ; UNCONDITIONAL BR
 8261
```

```
                                  PAGE
                           ;
                           ;      NON-NUMERIC IN EXPONENT
                           ;
                           !NON2
D8A3 C92B                         CMP     #'+'           ; IS IT PLUS?
D8A5 F006                         BEQ     !EPLUS         ; IF YES BR
D8A7 C92D                         CMP     #'-'           ; IS IT A MINUS?
D8A9 D007                         BNE     !NOTE          ; IF NOT, BR
                           ;
                           ;
                           !EMIN
D8AB 85EF                         STA     ESIGN          ; SAVE EXPONENT SIGN
                           !EPLUS
D8AD 2094DB                       JSR     !GETCHAR       ; GET CHARACTER
D8B0 90BA                         BCC     !EXP2          ; IF A #, GO PROCESS EXPONENT
                           ;
                           ;
                           ;
                           ;      E NOT PART OF OUR #
                           ;
                           !NOTE
D8B2 A5EC                         LDA     FRX            ; POINT TO 1 PAST E
D8B4 85F2                         STA     CIX            ; RESTORE CIX
                           ;
                           ;      FALL THRU TO EXIT
```

```
                                PAGE
                  ;
                  ;             WHOLE # HAS BEEN INPUTTED
                  ;
                  !EXIT
                  ;
                  ;             BACK UP ONE CHAR
                  ;
D8B6 C6F2                       DEC     CIX                ; DECREMENT INDEX
                  ;
                  ;
                  ;             CALCULATE POWER OF 10 = EXP - DIGITS RIGHT
                  ;             WHERE EXP = ENTERED EXPONENT (COMPLEMENT OF -)
                  ;                     + # DIGITS MORE THAN 9
                  ;
D8B8 A5ED                       LDA     EEXP               ; GET EXPONENT
D8BA A6F1                       LDX     DIGRT              ; GET # DIGITS RIGHT OF DECIMAL
D8BC 3005                       BMI     !EXIT1             ; NO DECIMAL POINT
D8BE F003                       BEQ     !EXIT1             ; # OF DIGITS AFTER D.P.=0
D8C0 38                         SEC                        ; GET EXP - DIGITS RIGHT AS NEW EXP
D8C1 E5F1                       SBC     DIGRT              ; X
                  ;
                  ;             SHIFT RIGHT ALGEBRAIC TO DIVIDE BY 2 = POWER OF 100
                  ;
                  !EXIT1
D8C3 48                         PHA
D8C4 2A                         ROLA                       ; SET CARRY WITH SIGN OF EXPONENT
D8C5 68                         PLA                        ; GET EXPONENT AGAIN
D8C6 6A                         RORA                       ; SHIFT RIGHT
D8C7 85ED                       STA     EEXP               ; SAVE POWER OF 100
D8C9 9003                       BCC     !EVEN              ; IF NO CARRY # EVEN
                  ;
D8CB 20EBDB                     JSR     NIBSHO             ; ELSE SHIFT 1 NIBBLE LEFT (MULT BY 10)
                  !EVEN
D8CE A5ED                       LDA     EEXP               ; ADD 40 FOR EXCESS 64 + 4 FOR NORM
D8D0 18                         CLC                        ; X
D8D1 6944                       ADC     #$44               ; X
D8D3 85D4                       STA     FR0                ; SAVE AS EXPONENT
                  ;
D8D5 2000DC                     JSR     NORM               ; NORMALIZE NUMBER
D8D8 B00B                       BCS     !IND2              ; IF CARRY SET, IT'S AN ERROR
                  ;
                  ;             SET MANTISSA SIGN
                  ;
D8DA A6EE                       LDX     NSIGN              ; IS SIGN OF # MINUS?
D8DC F006                       BEQ     !INDON             ; IF NOT, BR
                  ;
D8DE A5D4                       LDA     FR0                ; GET EXPONENT
D8E0 0980                       ORA     #$80               ; TURN ON MINUS # BIT
D8E2 85D4                       STA     FR0                ; SET IN FR0 EXP
                  !INDON
D8E4 18                         CLC                        ; CLEAR CARRY
                  !IND2
D8E5 60                         RTS
```

```
                                  PAGE
 8342                   *
 8343                   *         FPASC - CONVERT FLOATING POINT TO ASCII
 8344                   *
 8345                   *         ON ENTRY    FRO - # TO CONVERT
 8346                   *
 8347                   *         ON EXIT     INBUFF - POINTS TO START OF #
 8348                   *                     HIGH ORDER BIT OF LAST BYTE IS ON
 8349                   *
 8350                   *
 8351                   CVFASC
 8352                   FASC
 8353 D8E6 2051DA               JSR     INTLBF           ;SET INBUFF TO PT TO LBUFF
 8354                   ;
 8355 D8E9 A930                 LDA     #'0'             ; GET ASCII ZERO
 8356 D8EB 8D7F05               STA     LBUFF-1          ; PUT IN FROUNT OF LBUFF
 8357                   ;
 8358                   ;         TEST FOR E FORMAT REQUIRED
 8359                   ;
 8360 D8EE A5D4                 LDA     FRO              ; GET EXPONENT
 8361 D8F0 F028                 BEQ     !EXPO            ; IF EXP = 0, # = 0, SO BR
 8362 D8F2 297F                 AND     #$7F             ; AND OUT SIGH
 8363 D8F4 C93F                 CMP     #$3F             ; IS IT LESS THAN 3F
 8364 D8F6 9028                 BCC     !EFORM           ; IF YES, E FORMAT REQUIRED
 8365 D8F8 C945                 CMP     #$45             ; IF IT IS > 44
 8366 D8FA B024                 BCS     !EFORM           ; IF YES, E FORMAT REQUIRED
 8367                   *
 8368                   *         PROCESS NOT E FORMAT
 8369                   *
 8370 D8FC 38                   SEC                      ; SET CARRY
 8371 D8FD E93F                 SBC     #$3F             ; GET DECIMAL POSITION
 8372                   ;
 8373 D8FF 2070DC               JSR     !CVFRO           ; CONVERT FRO TO ASCII CHAR IN LBUFF
 8374                   ;
 8375 D902 20A4DC               JSR     !FNZERO          ; FIND LAST NON-ZERO CHARACTER
 8376 D905 0980                 ORA     #$80             ; TURN ON HIGH ORDER BIT FOR POINT END
 8377 D907 9D8005               STA     LBUFF,X          ; STORE IT BACK IN BUFFER
 8378                   ;
 8379 D90A AD8005               LDA     LBUFF            ; GET 1ST CHAR IN LBUFF
 8380 D90D C92E                 CMP     #'.'             ; IS IT DECIMAL?
 8381 D90F F003                 BEQ     !FN6             ; BR IF YES
 8382 D911 4C88D9               JMP     !FN5             ; ELSE JUMP
 8383                   !FN6
 8384 D914 20C1DC               JSR     !DECINB          ; DECIMAL INBUFF
 8385 D917 4C9CD9               JMP     !FN4             ; DO FINAL ADJUSTMENT
 8386
```

```
                                 PAGE
8387                   *
8388                   *         EXPONENT IS ZERO - # IS ZERO
8389                   *
8390                   *
8391                   !EXPO
8392 D91A A9B0                   LDA     #$80+$30          ; GET ASCII 0 WITH MSB = 1
8393 D91C 8D8005                 STA     LBUFF             ; PUT IN BUFFER
8394 D91F 60                     RTS
8395
```

```
                                   PAGE
 8396                  *
 8397                  *           PROCESS E FORMAT
 8398                  *
 8399                  !EFORM
 8400 D920 A901                    LDA      #1               ; GET DECIMAL POSITION
 8401 D922 2070DC                  JSR      !CVFRO           ; CONVERT FRO TO ASCII IN LBUFF
 8402                  ;
 8403 D925 20A4DC                  JSR      !FNZERO          ; GET RID OF TRAILING ZEROS
 8404 D928 E8                       INX                       ; INCR INDEX
 8405 D929 86F2                     STX      CIX              ; SAVE INDEX TO LAST CHAR
 8406                  ;
 8407                  ;           ADJUST EXPONENT
 8408                  ;
 8409 D92B A5D4                     LDA      FRO              ; GET EXPONENT
 8410 D92D 0A                       ASLA                      ; MULT BY 2 (GET RID OF SIGN TOO)
 8411 D92E 38                       SEC
 8412 D92F E980                     SBC      #$40*2           ; SUB EXCESS 64
 8413                  ;
 8414 D931 AE8005                   LDX      LBUFF            ; GET 1ST CHAR IN LBUFF
 8415 D934 E030                     CPX      #'0'             ; IS IT ASCII 0?
 8416 D936 F017                     BEQ      !EF1
 8417                  ;
 8418                  ;           PUT DECIMAL AFTER 1ST CHAR (IT'S AFTER 2ND NOW)
 8419                  ;
 8420 D938 AE8105                   LDX      LBUFF+1          ; SWITCH D.P. + 2ND DIGIT
 8421 D93B AC8205                   LDY      LBUFF+2          ; X
 8422 D93E 8E8205                   STX      LBUFF+2          ; X
 8423 D941 8C8105                   STY      LBUFF+1          ; X
 8424                  ;
 8425                  ;
 8426 D944 A6F2                     LDX      CIX              ; IF CIX POINTS TO D.P.
 8427 D946 E002                     CPX      #2               ; THEN INC
 8428 D948 D002                     BNE      !NOINC           ; X
 8429 D94A E6F2                     INC      CIX              ; X
 8430                  ;
 8431                  !NOINC
 8432 D94C 18                       CLC                       ; X
 8433 D94D 6901                     ADC      #1               ; X
 8434                  ;
 8435                  ;           CONVERT EXP TO ASCII
 8436                  ;
 8437                  !EF1
 8438 D94F 85ED                     STA      EEXP             ; SAVE EXPONENT
 8439 D951 A945                     LDA      #'E'             ; GET ASCII E
 8440 D953 A4F2                     LDY      CIX              ; GET POINTER
 8441 D955 209FDC                   JSR      !STCHAR          ; STORE CHARACTER
 8442 D958 84F2                     STY      CIX              ; SAVE INDEX
 8443                  ;
 8444                  ;
 8445 D95A A5ED                     LDA      EEXP             ; GET EXPONENT
 8446 D95C 100B                     BPL      !EPL             ; BR IF PLUS
 8447                  ;
 8448                  ;           EXPONENT IS MINUS - COMPLEMENT IT
 8449                  ;
```

```
D95E A900                LDA     #0              ; SUBTRACT FROM 0 TO COMPLEMENT
D960 38                  SEC                     ; X
D961 E5ED                SBC     EEXP            ; X
D963 85ED                STA     EEXP
                ;
D965 A92D                LDA     #'-'            ; GET A MINUS
D967 D002                BNE     !EF2
                ;
                !EPL
D969 A92B                LDA     #'+'            ; GET A PLUS
                !EF2
D96B 209FDC              JSR     !STCHAR         ; STORE A CHARACTER
                ;
D96E A200                LDX     #0              ; SET COUNTER FOR # OF TENS
D970 A5ED                LDA     EEXP            ; GET EXPONENT
                ;
                !EF3
D972 38                  SEC
D973 E90A                SBC     #10             ; SUBTRACT 10
D975 9003                BCC     'EF4            ; IF < 0, BRANCH
D977 E8                  INX                     ; INCR # OF 10'S
D978 D0F8                BNE     !EF3            ; BR UNCONDITIONAL
                ;
                !EF4
D97A 18                  CLC                     ; ADD BACK IN 10
D97B 690A                ADC     #10             ; X
D97D 48                  PHA                     ; SAVE
                ;
D97E 8A                  TXA                     ; GET # OF 10'S
D97F 209DDC              JSR     !STNUM          ; PUT 10'S IN EXP IN BUFFER
D982 68                  PLA                     ; GET REMAINDER
D983 0980                ORA     #$80            ; TURN ON HIGH ORDER BIT FOR PRINT END
D985 209DDC              JSR     'STNUM          ; PUT IN BUFFER
                ;
                ;        FINAL ADJUSTMENT
                ;
                'FN5
D988 AD8005              LDA     LBUFF           ; GET 1ST BYTE IN LBUFF (OUTPUT)
D98B C930                CMP     #'0'            ; IS IT ASCII 0?
D98D D00D                BNE     'FN4            ; IF NOT BR
                ;
                ;        INCREMENT INBUFF TO POINT TO NON-ZERO
                ;
D98F 18                  CLC                     ; ADD 1 TO INBUFF
D990 A5F3                LDA     INBUFF          ; X
D992 6901                ADC     #1              ; X
D994 85F3                STA     INBUFF          ; X
D996 A5F4                LDA     INBUFF+1        , X
D998 6900                ADC     #0              ; X
D99A 85F4                STA     INBUFF+1        ; X
                !FN4
D99C A5D4                LDA     FR0             ; GET EXPONENT OF #
D99E 1009                BPL     'FADONE         ; IF SIGN +, WE ARE DONE
                ;
D9A0 20C1DC              JSR     'DECINB         ; DECR INBUFF
```

```
 8505 D9A3 A000                   LDY      #0                  ; GET INDEX
 8506 D9A5 A92D                   LDA      #'-'                ; GET ASCII -
 8507 D9A7 91F3                   STA      (INBUFF),Y          ; SAVE - IN BUFFER
 8508                    ;
 8509                    !FADONE
 8510 D9A9 60                     RTS
 8511
```

```
                          PAGE
                *
                *         IFP - CONVERT INTEGER TO FLOATING POINT
                *
                *         ON ENTRY   FRO - CONTAINS INTEGER
                *
                *         ON EXIT    FRO - CONTAINS FLOATING POINT #
                *
                *
                CVIFP
                IFP
                ;
                ;         MOVE INTEGER AND REVERSE BYTES
                ;
D9AA A5D4                 LDA     FRO               ; GET INTEGER LOW
D9AC 85F8                 STA     ZTEMP4+1          ; SAVE AS INTEGER HIGH
D9AE A5D5                 LDA     FRO+1             ; GET INTEGER HIGH
D9B0 85F7                 STA     ZTEMP4            ; SAVE AS INTEGER LOW
                ;
D9B2 2044DA               JSR     ZFRO              ; CLEAR FRO
D9B5 F8                   SED                       ; SET DECIMAL MODE
                *
                *         DO THE CONVERT
                *
D9B6 A010                 LDY     #16               ; GET # BITS IN INTEGER
                !IFP1
D9B8 06F8                 ASL     ZTEMP4+1          ; SHIFT LEFT INTEGER LOW
D9BA 26F7                 ROL     ZTEMP4            ; SHIFT LEFT INTEGER HIGH
                                                    ; CARRY NOW SET IF THERE WAS A BIT
                ;
D9BC A203                 LDX     #3                ; BIGGEST INTEGER IS 3 BYTES
                !IFP2
                ;
                ;         DOUBLE # AND ADD IN 1 IF CARRY SET
                ;
D9BE B5D4                 LDA     FRO,X             ; GET BYTE
D9C0 75D4                 ADC     FRO,X             ; DOUBLE (ADDING IN CARRY FROM SHIFT)
D9C2 95D4                 STA     FRO,X             ; SAVE
D9C4 CA                   DEX                       ; DECREMENT COUNT OF FRO BYTES
D9C5 D0F7                 BNE     !IFP2             ; IF MORE TO DO, DO IT
                ;
D9C7 88                   DEY                       ; DECR COUNT OF INTEGER DIGITS
D9C8 D0EE                 BNE     !IFP1             ; IF MORE TO DO, DO IT
D9CA D8                   CLD                       ; CLEAR DECIMAL MODE
                ;
                ;         SET EXPONENT
                ;
D9CB A942                 LDA     #$42              ; INDICATE DECIMAL AFTER LAST DIGIT
D9CD 85D4                 STA     FRO               ; STORE EXPONENT
                ;
D9CF 4C00DC               JMP     NORM              ; NORMALIZE
                ;
```

```
                             PAGE
                     *
                     *       FPI - CONVERT FLOATING POINT TO INTEGER
                     *
                     *       ON ENTRY   FR0 - FLOATING POINT NUMBER
                     *
                     *       ON EXIT    FR0 - INTEGER
                     *
                     *
                     *       CC SET  CARRY CLEAR - NO ERROR
                     *               CARRY SET - ERROR
                     *
                     *
                     FPI
                     ;
                     ;       CLEAR INTEGER
                     ;
D9D2 A900                    LDA     #0               ; CLEAR INTEGER RESULT
D9D4 85F7                    STA     ZTEMP4
D9D6 85F8                    STA     ZTEMP4+1
                     ;
                     ;       CHECK EXPONENT
                     ;
D9D8 A5D4                    LDA     FR0              ; GET EXPONENT
D9DA 3066                    BMI     :ERVAL           ; IF SIGN OF FP# IS -, THEN ERROR
D9DC C943                    CMP     #$43             ; IS FP# TOO BIG TO BE INTEGER
D9DE B062                    BCS     :ERVAL           ; IF YES, THEN ERROR
D9E0 38                      SEC                      ; SET CARRY
D9E1 E940                    SBC     #$40             ; IS FP# LESS THAN 1?
D9E3 903F                    BCC     :ROUND           ; IF YES, THEN GO TEST FOR ROUND
                     ;
                     ;       GET # OF DIGITS TO CONVERT = (EXPONENT -40+1)*2
                     ;       (A CONTAINS EXPONENT -40)
                     ;       (CARRY SET)
                     ;
D9E5 6900                    ADC     #0               ; ADD IN CARRY
D9E7 0A                      ASLA                     ; MULT BY 2
D9E8 85F5                    STA     ZTEMP1           ; SAVE AS COUNTER
                     *
                     *       DO CONVERT
                     *
                     :FPI1
                     ;
                     ;       MULT INTEGER RESULT BY 10
                     ;
D9EA 205ADA                  JSR     :ILSHFT          ; GO SHIFT ONCE LEFT
D9ED B053                    BCS     :ERVAL           ; IF CARRY SET THEN # TO BIG
                     ;
D9EF A5F7                    LDA     ZTEMP4           ; SAVE INTEGER *2
D9F1 85F9                    STA     ZTEMP3           ; X
D9F3 A5F8                    LDA     ZTEMP4+1         ; X
D9F5 85FA                    STA     ZTEMP3+1         ; X
                     ;
D9F7 205ADA                  JSR     :ILSHFT          ; MULT BY *2
D9FA B046                    BCS     :ERVAL           ; # TO BIG
```

```
D9FC 205ADA                 JSR     !ILSHFT         ; MULT BY *2 (NOW * 8 IN ZTEMP4)
D9FF B041                   BCS     !ERVAL          ; BR IF # TO BIG
                    ;
DA01 18                     CLC                     ; ADD IN * 2 TO = *10
DA02 A5F8                   LDA     ZTEMP4+1        ; X
DA04 65FA                   ADC     ZTEMP3+1        ; X
DA06 85F8                   STA     ZTEMP4+1        ; X
DA08 A5F7                   LDA     ZTEMP4          ; X
DA0A 65F9                   ADC     ZTEMP3          ; X
DA0C 85F7                   STA     ZTEMP4          ; X
DA0E B032                   BCS     !ERVAL          ; IF CARRY SET ERROR
                    ;
                    ;
                    ;       ADD IN NEXT DIGIT
                    ;
DA10 20B9DC                 JSR     !GETDIG         ; GET DIGIT IN A
DA13 18                     CLC
DA14 65F8                   ADC     ZTEMP4+1        ; ADD IN DIGIT
DA16 85F8                   STA     ZTEMP4+1        ; X
DA18 A5F7                   LDA     ZTEMP4          ; X
DA1A 6900                   ADC     #0              ; X
DA1C B024                   BCS     !ERVAL          ; BR IF OVERFLOW
DA1E 85F7                   STA     ZTEMP4          ;X
                    ;
DA20 C6F5                   DEC     ZTEMP1          ; DEC COUNTER OF DIGITS TO DO
DA22 D0C6                   BNE     !FPI1           ; IF MORE TO DO, DO IT
                    ;
                    ;       ROUND IF NEEDED
                    ;
                    !ROUND
DA24 20B9DC                 JSR     !GETDIG         ; GET NEXT DIGIT IN A
DA27 C905                   CMP     #5              ; IS DIGIT <5?
DA29 900D                   BCC     !NR             ; IF YES, DON'T ROUND
DA2B 18                     CLC                     ; ADD IN 1 TO ROUND
DA2C A5F8                   LDA     ZTEMP4+1        ; X
DA2E 6901                   ADC     #1              ; X
DA30 85F8                   STA     ZTEMP4+1        ; X
DA32 A5F7                   LDA     ZTEMP4          ; X
DA34 6900                   ADC     #0              ; X
DA36 85F7                   STA     ZTEMP4          ; X
                    ;
                    ;       MOVE INTEGER TO FR0
                    ;
                    !NR
DA38 A5F8                   LDA     ZTEMP4+1        ; GET INTEGER LOW
DA3A 85D4                   STA     FR0             ; SAVE
DA3C A5F7                   LDA     ZTEMP4          ; GET INTEGER HIGH
DA3E 85D5                   STA     FR0+1           ; SAVE
                    ;
DA40 18                     CLC                     ; CLEAR CC FOR GOOD RETURN
DA41 60                     RTS
                    ;
                    ;
                    !ERVAL
DA42 38                     SEC                     ; SET CARRY FOR ERROR RETURN
```

```
 8673 DA43 60                      RTS
 8674
```

```
                                PAGE
                    *
                    *           ZFR0 - ZERO FR0
                    *
                    *           ZF1 - ZERO 6 BYTES AT LOC X
                    *
                    *           ZXLY - ZERO PAGE ZERO LOC X FOR LENGTH Y
                    *
                    ;
                    ZFR0
DA44 A2D4                       LDX     #FR0            ; GET POINTER TO FR1
                    ;
                    ZF1
DA46 A006                       LDY     #6              ; GET # OF BYTES TO CLEAR
                    ZXLY
DA48 A900                       LDA     #0              ; CLEAR A
                    !ZF2
DA4A 9500                       STA     0,X             ; CLEAR A BYTE
DA4C E8                         INX                     ; POINT TO NEXT BYTE
DA4D 88                         DEY                     ; DEC COUNTER
DA4E D0FA                       BNE     !ZF2            ; LOOP
DA50 60                         RTS
                    ;
                    ;
                    ;
                    ;
                    ;;          INTLBF - INIT LBUFF INTO INBUFF
                    ;
                    INTLBF
DA51 A905                       LDA     #LBUFF/256
DA53 85F4                       STA     INBUFF+1
DA55 A980                       LDA     #LBUFF&255
DA57 85F3                       STA     INBUFF
DA59 60                         RTS
```

```
                                PAGE
 8709                     *
 8710                     *       !ILSHFT - SHIFT INTEGER IN ZTEMP4 LEFT ONCE
 8711                     *
 8712                     ILSHFT
 8713                     !ILSHFT
 8714 DA5A 18                     CLC                          ; CLEAR CARRY
 8715 DA5B 26F8                   ROL      ZTEMP4+1            ; SHIFT LOW
 8716 DA5D 26F7                   ROL      ZTEMP4              ; SHIFT HIGH
 8717 DA5F 60                     RTS
 8718
```

```
                           PAGE     'FLOATING POINT ROUTINES'
 8719
```

```
                                 PAGE
 8720                  *
 8721                  *         FADD - FLOATING POINT ADD ROUTINE
 8722                  *                ADDS VALUES IN FR0 AND FR1
 8723                  *
 8724                  *         ON ENTRY   FR0 & FR1 - CONTAIN # TO ADD
 8725                  *
 8726                  *         ON EXIT    FR0 - RESULT
 8727                  *
 8728                  *
 8729                  *         FSUB - FLOATING POINT SUBTRACT ROUTINE
 8730                  *                SUBTRACTS FR1 FROM FR0
 8731                  *
 8732                  *         ON ENTRY   FR0 & FR1 - CONTAIN # TO SUBTRACT
 8733                  *
 8734                  *         ON EXIT    FR0 - RESULT
 8735                  *
 8736                  *         BOTH RETURN WITH CC SET:
 8737                  *                CARRY SET IF ERROR
 8738                  *                CARRY CLEAR IF NO ERROR
 8739                  *
 8740                  *
 8741                  FSUB
 8742 DA60 A5E0                  LDA     FR1              ; GET EXPONENT OF FR1
 8743 DA62 4980                  EOR     #$80             ; CHANGE SIGN OF MANTISSA
 8744 DA64 85E0                  STA     FR1              ; SAVE EXPONENT
 8745                  ;
 8746                  ;
 8747                  ;
 8748                  FADD
 8749                  !FRADD
 8750 DA66 A5E0                  LDA     FR1              ; GET EXPONENT FR1
 8751 DA68 297F                  AND     #$7F             ; TURN OFF MANTISSA SIGN BIT
 8752 DA6A 85F7                  STA     ZTEMP4           ; SAVE TEMPORARILY
 8753 DA6C A5D4                  LDA     FR0              ; GET EXPONENT FR0
 8754 DA6E 297F                  AND     #$7F             ; TURN OFF MANTISSA SIGN BIT
 8755 DA70 38                    SEC                      ; CLEAR CARRY
 8756 DA71 E5F7                  SBC     ZTEMP4           ; SUB EXPONENTS
 8757 DA73 1010                  BPL     !NSWAP           ; IF EXP(FR0)>= EXP(FR1), NO SWAP
 8758                  ;
 8759                  ;         SWAP FR0 AND FR1
 8760                  ;
 8761 DA75 A205                  LDX     #FMPREC          ; GET INDEX
 8762                  ;
 8763                  !SWAP
 8764 DA77 B5D4                  LDA     FR0,X            ; GET BYTE FROM FR0
 8765 DA79 B4E0                  LDY     FR1,X            ; GET BYTE FROM FR1
 8766 DA7B 95E0                  STA     FR1,X            ; PUT FR0 BYTE IN FR1
 8767 DA7D 98                    TYA                      ; GET FR1 BYTE
 8768 DA7E 95D4                  STA     FR0,X            ; PUT FR1 BYTE IN FR0
 8769 DA80 CA                    DEX                      ; DEC INDEX
 8770 DA81 10F4                  BPL     !SWAP            ; IF MORE TO DO, GO SWAP
 8771 DA83 30E1                  BMI     !FRADD           ; UNCONDITIONAL
 8772                  ;
 8773                  !NSWAP
```

```
8774 DA85 F007                  BEQ     !NALIGN          ; IF DIFFERENCE = 0, ALREADY ALIGNED
8775 DA87 C905                  CMP     #FMPREC          ; IS DIFFERENCE < # OF BYTES IN MANTISSA?
8776 DA89 B019                  BCS     !ADDEND          ; IF NOT, HAVE RESULT IN FRO
8777                    ;
8778                    ;
8779 DA8B 203EDC                JSR     RSHFT1           ; SHIFT TO ALIGN
8780                    ;
8781                    ;       TEST FOR LIKE SIGN OF MANTISSA
8782                    ;
8783                    !NALIGN
8784 DA8E F8                    SED                      ; SET DECIMAL MODE
8785 DA8F A5D4                  LDA     FRO              ; GET FRO EXPONENT
8786 DA91 45E0                  EOR     FR1              ; EOR WITH FR1 EXPONENT
8787 DA93 301E                  BMI     !SUB             ; IF SIGNS DIFFERENT - SUBTRACT
8788                                                     ; ELSE ADD
8789                    ;
8790                    ;       ADD FRO & FR1
8791                    ;
8792 DA95 A204                  LDX     #FMPREC-1        ; GET POINTER FOR LAST BYTE TO ADD
8793 DA97 18                    CLC                      ; CLEAR CARRY
8794                    !ADD1
8795 DA98 B5D5                  LDA     FROM,X           ; GET BYTE OF FRO
8796 DA9A 75E1                  ADC     FR1M,X           ; ADD IN BYTE OF FR1
8797 DA9C 95D5                  STA     FROM,X           ; STORE
8798 DA9E CA                    DEX                      ; DEC POINTER
8799 DA9F 10F7                  BPL     !ADD1            ; ADD NEXT BYTE
8800                    ;
8801 DAA1 D8                    CLD                      ; CLEAR DECIMAL MODE
8802 DAA2 B003                  BCS     !ADD2            ; IF THERE IS A CARRY, DO IT
8803                    !ADDEND
8804 DAA4 4C00DC                JMP     NORM             ; GO NORMALIZE
8805                    ;
8806                    ;       ADD IN FIND CARRY
8807                    ;
8808                    !ADD2
8809 DAA7 A901                  LDA     #1               ; GET 1 TIMES TO SHIFT
8810 DAA9 203ADC                JSR     RSHFTO           ; GO SHIFT
8811                    ;
8812 DAAC A901                  LDA     #01              ; GET CARRY
8813 DAAE 85D5                  STA     FROM             ; ADD IN CARRY
8814 DAB0 4C00DC                JMP     NORM
8815                    ;
8816                    ;       SUBTRACT FR1 FROM FRO
8817                    ;
8818                    !SUB
8819 DAB3 A204                  LDX     #FMPREC-1        ; GET POINTER TO LAST BYTE TO SUB
8820 DAB5 38                    SEC                      ; SET CARRY
8821                    ;
8822                    !SUB1
8823 DAB6 B5D5                  LDA     FROM,X           ; GET FRO BYTE
8824 DAB8 F5E1                  SBC     FR1M,X           ; SUB FR1 BYTE
8825 DABA 95D5                  STA     FROM,X           ; STORE
8826 DABC CA                    DEX                      ; DEC POINTER
8827 DABD 10F7                  BPL     !SUB1            ; SUB NEXT BYTE
8828                    ;
```

```
DABF 9004             BCC     !SUB2           ; IF THERE IS A BORROW DO IT
DAC1 D8               CLD                     ; CLEAR DECIMAL MODE
DAC2 4C00DC           JMP     NORM
                 ;
                 ;    TAKE COMPLIMENT SIGN
                 ;
                 !SUB2
DAC5 A5D4             LDA     FR0             ; GET EXPONENT
DAC7 4980             EOR     #$80            ; CHANGE SIGN OF MANTISSA
DAC9 85D4             STA     FR0             ; PUT IT BACK
                 ;
                 ;    COMPLIMENT MANTISSA
                 ;
DACB 38               SEC                     ; SET CARRY
DACC A204             LDX     #FMPREC-1       ; GET INDEX COUNTER
                 !SUB3
DACE A900             LDA     #0              ; GET ZERO
DAD0 F5D5             SBC     FR0M,X          ; COMPLEMENT BYTE
DAD2 95D5             STA     FR0M,X          ; STORE
DAD4 CA               DEX                     ; MORE TO DO
DAD5 10F7             BPL     !SUB3           ; BR IF YES
                 ;
DAD7 D8               CLD                     ; CLEAR DECIMAL MODE
DAD8 4C00DC           JMP     NORM            ; GO NORMALIZE
```

```
                                 PAGE
                  *
                  *              FPMULT - MULTIPLY FR0 BY FR1
                  *
                  *              ON ENTRY    # ARE IN FR0 AND FR1
                  *
                  *              ON EXIT     FR0 - CONTAINS PRODUCT
                  *              RETURN WITH CC SET
                  *                         CARRY SET IF ERROR
                  *                         CARRY CLEAR IF NO ERROR
                  *
                  *
                  *
                  FMUL
                  ;
                  ;              SET UP EXPONENT
                  ;
DADB A5D4                        LDA     FR0                ; GET EXP FR0
DADD F045                        BEQ     MEND3              ; IF = 0, DONE
DADF A5E0                        LDA     FR1                ; GET FR1 EXP
DAE1 F03E                        BEQ     MEND2              ; IF =0, ANSWER =0
                  ;
DAE3 20CFDC                      JSR     MDESUP             ; DO COMMON SET FOR EXPONENT
DAE6 38                          SEC                        ; SET CARRY
DAE7 E940                        SBC     #$40               ; SUB EXCESS 64
DAE9 38                          SEC                        ; SET CARRY TO ADD 1
DAEA 65E0                        ADC     FR1                ; ADD 1 + FR1 EXP TO FR0 EXP
DAEC 3038                        BMI     !EROV              ;IF - THEN OVERFLOW
                  ;
                  ;              FINISH MULTIPLY SET UP
                  ;
DAEE 20E0DC                      JSR     MDSUP              ; DO SET UP COMMON TO DIVIDE
                  ;
                  *
                  *              DO THE MULTIPLY
                  *
                  !FRM
                  ;
                  ;              GET # OF TIMES TO ADD IN MULTIPLICAND
                  ;
DAF1 A5DF                        LDA     FRE+FMPREC         ; GET LAST BYTE OF FRE
DAF3 290F                        AND     #$0F               ; AND OUT HIGH ORDER NIBBLE
DAF5 85F6                        STA     ZTEMP1+1           ; SET COUNTER FOR LOOP CONTROL
                  ;
                  ;              ADD IN FR1
                  ;
                  !FRM1
DAF7 C6F6                        DEC     ZTEMP1+1           ; DEC MULT COUNTER
DAF9 3006                        BMI     'FRM2              ; IF - THIS LOOP DONE
DAFB 2001DD                      JSR     FRA10              ; ADD FR1 TO FR0 (6 BYTES)
DAFE 4CF7DA                      JMP     'FRM1              ; REPEAT
                  ;
                  ;              GET # OF TIMES TO ADD IN MULTIPLICAND * 10
                  ;
                  !FRM2
```

```
DB01 A5DF                LDA     FRE+FMPREC      ; GET LAST BYTE OF FRE
DB03 4A                  LSRA                    ; SHIFT OUT LOW ORDER NIBBLE
DB04 4A                  LSRA                    ; X
DB05 4A                  LSRA                    ; X
DB06 4A                  LSRA                    ; X
DB07 85F6                STA     ZTEMP1+1        ; SAVE AS COUNTER
               ;
               ;         ADD IN FR2
               ;
               !FRM3
DB09 C6F6                DEC     ZTEMP1+1        ; DECREMENT COUNTER
DB0B 3006                BMI     !NXTB           ; IF -, DO NEXT BYTE
DB0D 2005DD              JSR     FRA20           ; ADD FR2 TO FR0 (6 BYTES)
DB10 4C09DB              JMP     !FRM3           ; REPEAT
               ;
               ;         SET UP FOR NEXT SET OF ADDS
               ;
               !NXTB
               ;
               ;         SHIFT FR0/FRE RIGHT ONE BYTE
               ;                 (THEY ARE CONTIGUOUS)
               ;
DB13 2062DC              JSR     RSHFOE          ;SHIFT FR0/FRE RIGHT
               ;
               ;         TEST FOR # OF BYTES SHIFTED
               ;
DB16 C6F5                DEC     ZTEMP1          ; DECREMENT LOOP CONTROL
DB18 D0D7                BNE     !FRM            ; IF MORE ADDS TO DO, DO IT
               ;
               ;         SET EXPONENT
               ;
               MDEND
DB1A A5ED                LDA     EEXP            ; GET EXPONENT
DB1C 85D4                STA     FR0             ; STORE AS FR0 EXP
               ;
               ;
               MEND1
DB1E 4C04DC              JMP     NORM1           ; NORMALIZE
               ;
               ;
               ;
               MEND2
DB21 2044DA              JSR     ZFR0            ; CLEAR FR0
               MEND3
DB24 18                  CLC                     ; CLEAR CARRY FOR GOOD RTN
DB25 60                  RTS
               ;
               ;
               ;
               !EROV
DB26 38                  SEC                     ; SET CARRY FOR ERROR ROUTINE
DB27 60                  RTS                     ; RETURN
```

```
                          PAGE
                *
                *         FPDIV - FLOATING POINT DIVIDE
                *
                *         ON ENTRY   FRO - DIVIDEND
                *                    FR1 - DIVISOR
                *
                *         ON EXIT    FRO - QUOTIENT
                *
                *         RETURNS WITH CC SET:
                *                 CARRY CLEAR - ERROR
                *                 CARRY SET - NO ERROR
                *
                *
                FDIV
                ;
                ;         DO DIVIDE SET YUP
                ;
DB28 A5E0                 LDA      FR1                 ; GET FR1 EXP
DB2A F0FA                 BEQ      !EROV               ; IF =0, THEN OVERFLOW
DB2C A5D4                 LDA      FRO                 ; GET EXPONENT FRO
DB2E F0F4                 BEQ      MEND3               ; IF = 0, THEN DONE
                ;
DB30 20CFDC               JSR      MDESUP              ; DO COMMOM PART OF EXP SET UP
                ;
DB33 38                   SEC
DB34 E5E0                 SBC      FR1                 ; SUB FR1 EXP FROM FRO EX
DB36 18                   CLC
DB37 6940                 ADC      #$40                ; ADD IN EXCESS 64
DB39 30EB                 BMI      !EROV               ; IF MINUS THEN OVERFLOW
                ;
DB3B 20E0DC               JSR      MDSUP               ; DO SETUP COMMON FOR MULT
DB3E E6F5                 INC      ZTEMP1              ;LOOP 1 MORE TIME FOR DIVIDE
DB40 4C4EDB               JMP      !FRD1               ; SKIP SHIFT 1ST TIME THROUGH
                ;
     00D9       QTEMP     EQU      FRO+FMPREC
                !NXTQ
                ;
                ;         SHIFT FRO/FRE LEFT ONE BYTE
                ;                  (THEY ARE CONTIGUOUS)
                ;
DB43 A200                 LDX      #0                  ; GET POINTER TO BYTE TO MOVE
                !NXTQ1
DB45 B5D5                 LDA      FRO+1,X             ; GET BYTE
DB47 95D4                 STA      FRO,X               ; MOVE IT LEFT ONE BYTE
                ;
DB49 E8                   INX                          ; POINT TO NEXT BYTE
DB4A E00C                 CPX      #FMPREC*2+2         ; HAVE WE DONE THEM ALL?
DB4C D0F7                 BNE      !NXTQ1              ; IF NOT, BRANCH
                *
                *         DO DIVIDE
                *
                !FRD1
                ;
                ;         SUBTRACT FR2 (DIVISOR * 2) FROM FRE (DIVIDEND)
```

```
                  ;
                  ;
DB4E A005                   LDY     #FMPREC          ; SET LOOP CONTROL
DB50 38                     SEC                      ; SET CARRY
DB51 F8                     SED                      ; SET DECIMAL MODE
                  !FRS2
DB52 B9DA00                 LDA     FRE,Y            ; GET A BYTE FROM FRE
DB55 F9E600                 SBC     FR2,Y            ; SUB FR2
DB58 99DA00                 STA     FRE,Y            ; STORE RESULT
DB5B 88                     DEY                      ; DEC COUNTER
DB5C 10F4                   BPL     !FRS2            ; BR IF MORE TO DO
DB5E D8                     CLD                      ; CLEAR DECIMAL MODE
                  ;
DB5F 9004                   BCC     !FAIL            ; IF RESULT <0 (FRE < FR2) BR
                  ;
DB61 E6D9                   INC     QTEMP            ; INCR # TIMES SUB (QUOTINT)
                  ;
DB63 D0E9                   BNE     !FRD1            ; SUB AGAIN
                  ;
                  ;         SUBTRACT OF FR2 DIDN'T GO
                  ;
                  !FAIL
DB65 200FDD                 JSR     FRA2E            ; ADD FR2 BACK TO FR0
                  ;
                  ;         SHIFT LAST BYTE OF QUOTIENT ONE NIBBLE LEFT
                  ;
DB68 06D9                   ASL     QTEMP            ; SHIFT 4 BITS LEFT
DB6A 06D9                   ASL     QTEMP            ; X
DB6C 06D9                   ASL     QTEMP            ; X
DB6E 06D9                   ASL     QTEMP            ; X
                  !FRD2
                  ;
                  ;         SUBTRACT FR1 (DIVISOR) FROM FRE (DIVIDEND)
                  ;
                  ;
DB70 A005                   LDY     #FMPREC          ; SET LOOP CONTROL
DB72 38                     SEC                      ; SET CARRY
DB73 F8                     SED                      ; SET DECIMAL MODE
                  !FRS1
DB74 B9DA00                 LDA     FRE,Y            ; GET A BYTE FROM FRE
DB77 F9E000                 SBC     FR1,Y            ; SUB FR1
DB7A 99DA00                 STA     FRE,Y            ; STORE RESULT
DB7D 88                     DEY
DB7E 10F4                   BPL     !FRS1            ; BR IF MORE TO DO
DB80 D8                     CLD                      ; CLEAR DECIMAL MODE
                  ;
DB81 9004                   BCC     !FAIL2           ; IF RESULT < 0 (FRE < FR1) BR
                  ;
DB83 E6D9                   INC     QTEMP            ; INCR # TIMES SUB (QUOTIENT)
                  ;
DB85 D0E9                   BNE     !FRD2            ; SUB AGAIN
                  ;
                  ;         SUBTRACT OF FR1 DIDN'T GO
                  ;
                  !FAIL2
```

```
                               BNE     !FRD2            ; SUB AGAIN
 9066                  ;
 9067                  ;       SUBTRACT OF FR1 DIDN'T GO
 9068                  ;
 9069                  !FAIL2
```

```
 9070 DB87 2009DD          JSR     FRA1E            ; ADD FR1 BACK TO FRO
 9071                  ;
 9072 DB8A C6F5            DEC     ZTEMP1           ; DEC LOOP CONTROL
 9073 DB8C D0B5            BNE     !NXTQ            ; GET NEXT QUIOTIENT BYTE
 9074                  ;
 9075 DB8E 2062DC          JSR     RSHFOE           ;SHIFT RIGHT FRO/FRE TO CLEAR EXP
 9076 DB91 4C1ADB          JMP     MDEND            ; JOIN MULT END UP CODE
 9077
```

```
                                PAGE
                    *
                    *           !GETCHAR - TEST INPUT CHARACTER
                    *
                    *           ON ENTRY    INBUFF - POINTS TO BUFFER WITH INPUT
                    *                       CIX - POINTS TO CHAR IN BUFFER
                    *
                    *           ON EXIT     CIX - POINTS TO NEXT CHAR
                    *                       CC - CARRY CLEAR IF CHAR IS NUMBER
                    *                            CARRY SET IF CHAR NOT NUMBER
                    *
                    !GETCHAR
DB94 20AFDB                     JSR     TSTNUM          ; GO TEST FOR NUMBER
DB97 A4F2                       LDY     CIX             ; GET CHARACTER INDEX
DB99 9002                       BCC     !GCHR1          ; IF CHAR = NUM, SKIP
                    ;
DB9B B1F3                       LDA     (INBUFF),Y      ; GET CHARACTER
                    ;
                    !GCHR1
DB9D C8                         INY                     ; POINT TO NEXT CHAR
DB9E 84F2                       STY     CIX             ; SAVE INDEX
DBA0 60                         RTS
                    ;
                    ;SKPBLANK-SKIP BLANKS
                    ;           STARTS AT CIX AND SCANS FOR NON BLANKS
                    ;
                    SKBLANK
                    SKPBLANK
DBA1 A4F2                       LDY     CIX             ;GET CIX
DBA3 A920                       LDA     #$20            ;GET A BLANK
                    ;
DBA5 D1F3           !SB1        CMP     (INBUFF),Y      ;IS CHAR A BLANK
DBA7 D003                       BNE     !SBRTS          ;BR IF NOT
DBA9 C8                         INY                     ;INC TO NEXT
DBAA D0F9                       BNE     !SB1            ;GO TEST
                    ;
DBAC 84F2           !SBRTS      STY     CIX             ;SET NON BLANK INDEX
DBAE 60                         RTS                     ;RETURN
                    ;
                    ; TSTNUM-TEST CHAR AT CIX FOR NUM
                    ;           - RTNS CARRY SET IF NUM
                    TSTNUM
DBAF A4F2                       LDY     CIX             ;GET INDEX
DBB1 B1F3                       LDA     (INBUFF),Y      ;AND GET CHAR
DBB3 38                         SEC
DBB4 E930                       SBC     #$30            ;SUBTRACT ASCLT ZERO
DBB6 9018                       BCC     !TSNFAIL        ;BR CHAR<ASCLT ZERO
DBB8 C90A                       CMP     #$0A            ;TEST GT ASCLT 9
DBBA 60                         RTS                     ;DONE
```

```
                                PAGE
9127                 *
9128                 *          !TSTCHAR - TEST TO SEE IF THIS CAN BE A NUMBER
9129                 *
9130                 *          ON EXIT    CC - CARRY SET IF NOT A #
9131                 *                          CARRY CLEAR IF A #
9132                 *
9133                 !TSTCHAR
9134 DBBB A5F2                  LDA     CIX               ; GET INDEX
9135 DBBD 48                    PHA                       ; SAVE IT
9136 DBBE 2094DB                JSR     !GETCHAR          ; GET CHAR
9137 DBC1 901F                  BCC     !RTPASS           ; IF = #8 RETURN PASS
9138                 ;
9139 DBC3 C92E                  CMP     #'.'              ; IF = D.P., OK SO FAR
9140 DBC5 F014                  BEQ     !TSTN
9141 DBC7 C92B                  CMP     #'+'              ; IF = +8 OK SO FAR
9142 DBC9 F007                  BEQ     !TSTN1
9143 DBCB C92D                  CMP     #'-'              ; IF = -8 OK SO FAT
9144 DBCD F003                  BEQ     !TSTN1
9145                 ;
9146                 ;
9147                 !RTFAIL
9148 DBCF 68                    PLA                       ; CLEAR STACK
9149 DBD0 38         !TSNFAIL   SEC                       ;SET FAIL
9150 DBD1 60                    RTS
9151                 ;
9152                 ;
9153                 !TSTN1
9154 DBD2 2094DB                JSR     !GETCHAR          ; GET CHAR
9155 DBD5 900B                  BCC     !RTPASS           ; IF #, RETURN PASS
9156 DBD7 C92E                  CMP     #'.'              ; IS IT D.P.
9157 DBD9 D0F4                  BNE     !RTFAIL           ; IF NOT, RETURN FAIL
9158                 !TSTN
9159 DBDB 2094DB                JSR     !GETCHAR          ; ELSE GET NEXT CHAR
9160 DBDE 9002                  BCC     !RTPASS           ; IF #, RETURN PASS
9161 DBE0 B0ED                  BCS     !RTFAIL           ; ELSE, RETURN FAIL
9162                 ;
9163                 ;
9164                 !RTPASS
9165 DBE2 68                    PLA                       ; RESTORE CIX
9166 DBE3 85F2                  STA     CIX               ; X
9167 DBE5 18                    CLC                       ; CLEAR CARRY
9168 DBE6 60                    RTS                       ; RETURN PASS
9169
```

```
                                PAGE
                          *
                          *     NIBSHO - SHIFT FRO ONE NIBBLE LEFT
                          *
                          *     NIBSH2 - SHIFT FR2 ONE NIBBLE LEFT
                          *
                          NIBSH2
     DBE7 A2E7                  LDX     #FR2+1          ; POINT TO 1ST MANTISSA BYTE OF FR2
     DBE9 D002                  BNE     !NIB1
                          ;
                          NIBSHO
     DBEB A2D5                  LDX     #FROM           ; POINT TO MANTISSA OF FRO
                          !NIB1
     DBED A004                  LDY     #4              ; GET # OF BITS TO SHIFT
                          !NIBS
     DBEF 18                    CLC                     ; CLEAR CARRY
     DBF0 3604                  ROL     4,X             ; ROLL
     DBF2 3603                  ROL     3,X             ; X
     DBF4 3602                  ROL     2,X             ; X
     DBF6 3601                  ROL     1,X             ; X
     DBF8 3600                  ROL     0,X             ; X
     DBFA 26EC                  ROL     FRX             ; SAVE SHIFTED NIBBLE
                          ;
     DBFC 88                    DEY                     ; DEC COUNT
     DBFD D0F0                  BNE     !NIBS           ; IF NOT = 0, REPEAT
     DBFF 60                    RTS
```

```
                             PAGE
                    *
                    *        NORM - NORMMALIZE FLOATING POINT NUMBER
                    *
                    NORM
DC00 A200                    LDX     #0                ; GET ZERO
DC02 86DA                    STX     FRO+FPREC         ; FOR ADD NORM SHIFT IN A ZERO
                    NORM1
DC04 A204                    LDX     #FMPREC-1         ; GET MAX # OF BYTES TO SHIFT
DC06 A5D4                    LDA     FRO               ; GET EXPONENT
DC08 F02E                    BEQ     !NDONE            ; IF EXP=0, # =0
                    !NORM
DC0A A5D5                    LDA     FROM              ; GET 1ST BYTE OF MANTISSA
DC0C D01A                    BNE     !TSTBIG           ; IF NOT = 0 THEN NO SHIFT
                    ;
                    ;        SHIFT 1 BYTE LEFT
                    ;
DC0E A000                    LDY     #0                ; GET INDEX FOR 1ST MOVE BYTE
                    !NSH
DC10 B9D600                  LDA     FROM+1,Y          ; GET MOVE BYTE
DC13 99D500                  STA     FROM,Y            ; STORE IT
DC16 C8                      INY
DC17 C005                    CPY     #FMPREC           ; ARE WE DONE
DC19 90F5                    BCC     !NSH              ; IF NOT SHIFT AGAIN
                    ;
                    ;
                    ;        DECREMENT EXPONENT
                    ;
DC1B C6D4                    DEC     FRO               ;DECRIMENT EXPONENT
                    ;
DC1D CA                      DEX                       ; DEC COUNTER
DC1E D0EA                    BNE     !NORM             ; DO AGAIN IF NEEDED
                    ;
                    ;
                    ;
DC20 A5D5                    LDA     FROM              ; IS MANTISSA STILL 0
DC22 D004                    BNE     !TSTBIG           ; IF NOT, SEE IF TO BIG
DC24 85D4                    STA     FRO               ; ELSE ZERO EXP
DC26 18                      CLC
DC27 60                      RTS
                    ;
                    !TSTBIG
DC28 A5D4                    LDA     FRO               ; GET EXPONENT
DC2A 297F                    AND     #$7F              ; AND OUT SIGN BIT
DC2C C971                    CMP     #49+64            ; IS IT < 49+64?
DC2E 9001                    BCC     !TSTUND           ; IF YES, TEST UNDERFLOW
DC30 60                      RTS                       ; SO RETURN
                    !TSTUND
DC31 C90F                    CMP     #-49+64           ; IS IT >=-49+64?
DC33 B003                    BCS     !NDONE            ; IF YES, WE ARE DIONE
DC35 2044DA                  JSR     ZFRO              ; ELSE # IS ZERO
                    ;
                    !NDONE
DC38 18                      CLC                       ; CLEAR CARRY FOR GOOD RETURN
DC39 60                      RTS
```

```
 9250
```

```
                          PAGE
                *
                *         RSHFT0 - SHIFT FR0 RIGHT/INCR EXPONENT
                *
                *         RSHFT1 - SHIFT FR1 RIGHT/INCR EXPONENT
                *
                *         ON ENTRY   A - # OF PLACES TO SHIFT
                *
                *
                RSHFT0
DC3A A2D4                 LDX     #FR0              ; POINT TO FR0
DC3C D002                 BNE     !RSH
                ;
                RSHFT1
DC3E A2E0                 LDX     #FR1              ; POINT TO FR1
                ;
                !RSH
DC40 86F9                 STX     ZTEMP3            ; SAVE FR POINTER
DC42 85F7                 STA     ZTEMP4            ; SAVE # OF BYTES TO SHIFT AS COUNTER
DC44 85F8                 STA     ZTEMP4+1          ; SAVE FOR LATER
                ;
                !RSH2
DC46 A004                 LDY     #FMPREC-1         ; GET # OF BYTES TO MOVE EACH LOOP
                !RSH1
DC48 B504                 LDA     4,X               ; GET CHAR
DC4A 9505                 STA     5,X               ; STORE CHAR
DC4C CA                   DEX                       ; POINT TO NEXT BYTE
DC4D 88                   DEY                       ; DEC LOOP CONTROL
DC4E D0F8                 BNE     !RSH1             ; IF MORE TO MOVE, DO IT
DC50 A900                 LDA     #0                ; GET 1ST BYTE
DC52 9505                 STA     5,X               ; STORE IT
                ;
DC54 A6F9                 LDX     ZTEMP3            ; GET FR POINTER
DC56 C6F7                 DEC     ZTEMP4            ; DO WE NEED TO SHIFT AGAIN?
DC58 D0EC                 BNE     !RSH2             ; IF YES, DO IT
                ;
                ;         FIX EXPONENT
                ;
DC5A B500                 LDA     0,X               ; GET EXPONENT
DC5C 18                   CLC
DC5D 65F8                 ADC     ZTEMP4+1          ; SUB # OF SHIFTS
DC5F 9500                 STA     0,X               ; SAVE NEW EXPONENT
DC61 60                   RTS
```

```
                                   PAGE
9294                     *
9295                     *         RSHF0E- SHIFT FR0/FRE 1 BYTE RIGHT
9296                     *         (THEY ARE CONTIGUOUS)
9297                     *
9298                     RSHF0E
9299 DC62 A20A                     LDX     #FMPREC*2           ; GET LOOP CONTROL
9300                     ;
9301                     !NXTB1
9302 DC64 B5D4                     LDA     FR0,X               ; GET A BYTE
9303 DC66 95D5                     STA     FR0+1,X             ; MOVE IT OVER 1
9304                     ;
9305 DC68 CA                       DEX                         ; DEC COUNTER
9306 DC69 10F9                     BPL     !NXTB1              ; MOVE NEXT BYTE
9307 DC6B A900                     LDA     #0                  ; GET ZERO
9308 DC6D 85D4                     STA     FR0                 ; SHIFT IT IN
9309 DC6F 60                       RTS
9310
```

```
                          PAGE
                 *
                 *        !CVFRO - CONVERT EACH BYTE IN FRO TO 2 CHARACTERS
                 *                 IN LBUFF
                 *
                 *        ON ENTRY   A - DECIMAL POINT POSITION
                 *
                 *
                 !CVFRO
DC70 85F7                 STA     ZTEMP4            ; SAVE DECIMAL POSITION
                 ;
DC72 A200                 LDX     #0                ; SET INDEX INTO FROM
DC74 A000                 LDY     #0                ; SET INDEX INTO OUTPUT LINE (LBUFF)
                 ;
                 ;        CONVERT A BYTE
                 ;
                 !CVBYTE
DC76 2093DC               JSR     !TSTDP            ; PUT IN D.P. NOW?
                 !CVB1
DC79 38                   SEC                       ; DECREMENT DECIMAL POSITION
DC7A E901                 SBC     #1                ; X
DC7C 85F7                 STA     ZTEMP4            ; SAVE IT
                 ;
                 ;        DO 1ST DIGIT
                 ;
DC7E B5D5                 LDA     FROM,X            ; GET FROM FRO
DC80 4A                   LSRA                      ; SHIFT OUT LOW ORDER BITS
DC81 4A                   LSRA                      ; TO GET 1ST DIGIT
DC82 4A                   LSRA                      ; X
DC83 4A                   LSRA                      ; X
DC84 209DDC               JSR     !STNUM            ; GO PUT # IN BUFFER
                 ;
                 ;        DO SECOND DIGIT
                 ;
DC87 B5D5                 LDA     FROM,X            ; GET NUMBER FROM FRO
DC89 290F                 AND     #$0F              ; AND OUT HIGH ORDER BITS
DC8B 209DDC               JSR     !STNUM            ; GO PUT # IN BUFFER
                 ;
DC8E E8                   INX                       ; INCR FRO POINTER
DC8F E005                 CPX     #FMPREC           ; DONE LAST FRO BYTE?
DC91 90E3                 BCC     !CVBYTE           ; IF NOT, MORE TO DO
                 ;
                 ;        PUT IN DECIMAL POINT NOW?
                 ;
                 !TSTDP
DC93 A5F7                 LDA     ZTEMP4            ; GET DECIMAL POSITION
DC95 D005                 BNE     !TST1             ; IF NOT = 0 RTN
DC97 A92E                 LDA     #'.'              ; GET ASCII DECIMAL POINT
DC99 209FDC               JSR     !STCHAR           ; PUT D.P. IN BUFFER
                 !TST1
DC9C 60                   RTS
```

```
                                  PAGE
 9362                      *
 9363                      *      !STNUM - PUT ASCII NUMBER IN LBUFF
 9364                      *
 9365                      *      ON ENTRY   A - DIGIT TO BE CONVERTED TO ASCII
 9366                      *                     AND PUT IN LBUFF
 9367                      *                 Y - INDEX IN LBUFF
 9368                      *
 9369                      *      !STCHAR - STORES CHAR IN A IN LBUFF
 9370                      *
 9371                      !STNUM
 9372 DC9D 0930                   ORA     #$30              ; CONVERT TO ASCII
 9373                      !STCHAR
 9374 DC9F 998005                 STA     LBUFF,Y           ; PUT IN LBUFF
 9375 DCA2 C8                     INY                       ; INCR LBUFF POINTER
 9376 DCA3 60                     RTS
 9377
```

```
                                    PAGE
 9378                   *
 9379                   *           !FNZERO - FIND LAST NON-ZERO CHAR IN LBUFF
 9380                   *
 9381                   *           ON EXIT     A - LAST CHAR
 9382                   *                       X - POINT TO LAST CHAR
 9383                   *
 9384                   !FNZERO
 9385 DCA4 A20A                     LDX     #10             ; POINT TO LAST CHAR IN LBUFF
 9386                   ;
 9387                   !FN3
 9388 DCA6 BD8005                   LDA     LBUFF,X         ; GET THE CHARACTER
 9389 DCA9 C92E                     CMP     #'.'            ; IS IT DECIMAL?
 9390 DCAB F007                     BEQ     !FN1            ; IF YES, BR
 9391 DCAD C930                     CMP     #'0'            ; IS IT ZERO?
 9392 DCAF D007                     BNE     !FN2            ; IF NOT, BR
 9393 DCB1 CA                       DEX                     ; DECREMENT INDEX
 9394 DCB2 D0F2                     BNE     !FN3            ; UNCONDITIONAL BR
 9395                   ;
 9396      0000                     .IF     DBVER
 9397                               BRK                     ; SHOULD NEVER GET HERE
 9398                               .ENDIF
 9399                   ;
 9400                   !FN1
 9401 DCB4 CA                       DEX                     ; DECREMENT BUFFER INDEX
 9402 DCB5 BD8005                   LDA     LBUFF,X         ; GET LAST CHAR
 9403                   !FN2
 9404 DCB8 60                       RTS
 9405
```

```
                                    PAGE
 9406                       *
 9407                       *       !GETDIG - GET NEXT DIGIT FROM FRO
 9408                       *
 9409                       *       ON ENTRY   FRO - #
 9410                       *
 9411                       *       ON EXIT    A - DIGIT
 9412                       *
 9413                       *
 9414                       !GETDIG
 9415 DCB9 20EBDB                   JSR     NIBSHO           ; SHIFT FRO LEFT ONE NIBBLE
 9416                       ;
 9417 DCBC A5EC                     LDA     FRX              ; GET BYTE CONTAINING SHIFTED NIBBLE
 9418 DCBE 290F                     AND     #$0F             ; AND OUT HIGH ORDER NIBBLE
 9419 DCC0 60                       RTS
 9420
```

```
                       PAGE
               *
               *       !DECINB - DECREMENT INBUFF
               *
               !DECINB
DCC1 38                SEC                       ; SUBTRACT ONE FROM INBUFF
DCC2 A5F3              LDA     INBUFF            ; X
DCC4 E901              SBC     #1                ; X
DCC6 85F3              STA     INBUFF            ; X
DCC8 A5F4              LDA     INBUFF+1          ; X
DCCA E900              SBC     #0                ; X
DCCC 85F4              STA     INBUFF+1          ; X
DCCE 60                RTS
```

```
                          PAGE
                    *
                    *     MDESUP - COMMON SET UP FOR MULTPLY & DIVIDE EXPONENT
                    *
                    *     ON EXIT    FR1 - FR1 EXP WITH OUT SIGN
                    *                A - FR0 EXP WITHOUT SIGN
                    *                FRSIGN - SIGN FOR QUOTIENT
                    *
                    MDESUP
DCCF A5D4                 LDA     FR0              ; GET FR0 EXPONENT
DCD1 45E0                 EOR     FR1              ; GET FR1 EXPONENT
DCD3 2980                 AND     #$80             ; AND OUT ALL BUT SIGN BIT
DCD5 85EE                 STA     FRSIGN           ; SAVE SIGN
                    ;
DCD7 06E0                 ASL     FR1              ; SHIFT OUT SIGN IN FR1 EXP
DCD9 46E0                 LSR     FR1              ; RESTORE FR1 EXP WITHOUT SIGN
DCDB A5D4                 LDA     FR0              ; GET FR0 EXP
DCDD 297F                 AND     #$7F             ; AND OUT SIGN BIT
DCDF 60                   RTS
```

```
                                       PAGE
 9453                          *
 9454                          *       MDSUP - COMMON SET UP FOR MULTIPLY AND DIVIDE
 9455                          *
 9456                          *       ON ENTRY   A - EXPONENT
 9457                          *                 CC - SET BY ADD OR SUB TO GET A
 9458                          *
 9459                          *
 9460                          MDSUP
 9461  DCE0  05EE                      ORA    FRSIGN          ; OR IN SIGN BIT
 9462  DCE2  85ED                      STA    EEXP            ; SAVE EXPONENT FOR LATER
 9463  DCE4  A900                      LDA    #0              ; CLEAR A
 9464  DCE6  85D4                      STA    FR0             ; CLEAR FR0 EXP
 9465  DCE8  85E0                      STA    FR1             ; CLEAR FR0 EXP
 9466                          ;
 9467                          ;
 9468  DCEA  2028DD                    JSR    MVFR12          ; MOVE FR1 TO FR2
 9469                          ;
 9470  DCED  20E7DB                    JSR    NIBSH2          ; SHIFT FR2 1 NIBBLE LEFT
 9471  DCF0  A5EC                      LDA    FRX             ; GET SHIFTED NIBBLE
 9472  DCF2  290F                      AND    #$0F            ; AND OUT HIGH ORDER NIBBLE
 9473  DCF4  85E6                      STA    FR2             ; STORE TO FINISH SHIFT
 9474                          ;
 9475  DCF6  A905                      LDA    #FMPREC         ; SET LOOP CONTROL
 9476  DCF8  85F5                      STA    ZTEMP1          ; X
 9477                          ;
 9478  DCFA  2034DD                    JSR    MVFR0E          ; MOVE FR0 TO FRE
 9479  DCFD  2044DA                    JSR    ZFR0            ; CLEAR FR0
 9480                          ;
 9481  DD00  60                        RTS
 9482                          ;
 9483
```

```
                                PAGE
                     *
                     *          FRA10 - ADD FR1 TO FR0 (6 BYTES)
                     *
                     *          FRA20 - ADD FR2 TO FR0 (6 BYTES)
                     *
                     *          FRA1E - ADD FR1 TO FRE
                     *
                     *          FRA2E - ADD FR2 TO FRE
                     *
                     FRA10
      DD01 A2D9                 LDX     #FR0+FMPREC         ; POINT TO LAST BYTE OF SUM
      DD03 D006                 BNE     !F1
                     ;
                     FRA20
      DD05 A2D9                 LDX     #FR0+FMPREC
      DD07 D008                 BNE     !F2
                     ;
                     FRA1E
      DD09 A2DF                 LDX     #FRE+FMPREC
                     !F1
      DD0B A0E5                 LDY     #FR1+FMPREC
      DD0D D004                 BNE     !FRA
                     FRA2E
      DD0F A2DF                 LDX     #FRE+FMPREC
                     !F2
      DD11 A0EB                 LDY     #FR2+FMPREC
                     ;
                     ;
                     !FRA
      DD13 A905                 LDA     #FMPREC             ; GET VALUE FOR LOOP CONTROL
      DD15 85F7                 STA     ZTEMP4              ; SET LOOP CONTROL
      DD17 18                   CLC                         ; CLEAR CARRY
      DD18 F8                   SED                         ; SET DECIMAL MODE
                     !FRA1
      DD19 B500                 LDA     0,X                 ; GET 1ST BYTE OF
      DD1B 790000               ADC     0,Y                 ; ADD
      DD1E 9500                 STA     0,X                 ; STORE
      DD20 CA                   DEX                         ; POINT TO NEXT BYTE
      DD21 88                   DEY                         ; POINT TO NEXT BYTE
      DD22 C6F7                 DEC     ZTEMP4              ; DEC COUNTER
      DD24 10F3                 BPL     !FRA1               ; IF MORE TO DO, DO IT
      DD26 D8                   CLD                         ; CLEAR DECIMAL MODE
      DD27 60                   RTS
```

```
                             PAGE
                   *
                   *         MVFR12 - MOVE FR1 TO FR2
                   *
                   MVFR12
DD28 A005                    LDY     #FMPREC          ; SET COUNTER
                   !MV2
DD2A B9E000                  LDA     FR1,Y            ; GET A BYTE
DD2D 99E600                  STA     FR2,Y            ; STORE IT
                   ;
DD30 88                      DEY                      ; DEC COUNTER
DD31 10F7                    BPL     !MV2             ; IF MORE TO MOVE, DO IT
DD33 60                      RTS
                   ;
                   ;
                   ;
                   ;
                   *
                   *         MVFR0E - MOVE FR0 TO FRE
                   *
                   MVFR0E
DD34 A005                    LDY     #FMPREC
                   !MV1
DD36 B9D400                  LDA     FR0,Y
DD39 99DA00                  STA     FRE,Y
                   ;
DD3C 88                      DEY
DD3D 10F7                    BPL     !MV1
DD3F 60                      RTS

```

```
PAGE   292    SHEP  ATARI EDITOR
POLYNOMAL EVALUATION

                                PAGE    'POLYNOMAL EVALUATION'
  9557                  *       Y=A(0)+A(1)*X+A(2)*X**2+. . +A(N)*X**N, N>0
  9558                  *        =((. . (A(N)*X+A(N-1))*X+. . +A(2))*X+A(1))*X+A(0)
  9559                  *       INPUT: X IN FR0, N+1 IN A-REG
  9560                  *       REG (X,Y)->A(N). . A(0)
  9561                  *       OUTPUT  Y IN FR0
  9562                  *       USES FPTR2, PLYCNT, PLYARG
  9563                  *       CALLS FSTOR, FMOVE, FLD1R, FADD, FMUL
  9564 DD40 BE0000    PLYEVL  STX     FPTR2               ;SAVE POINTER TO COEFF'S
ERROR UNDEFINED  ***********************@
  9565 DD43 8C0100            STY     FPTR2+1
ERROR UNDEFINED  *************************@
  9566 DD46 85EF              STA PLYCNT
  9567 DD48 A200              LDX     #PLYARG&$FF
ERROR UNDEFINED  *****************************@
  9568 DD4A A000              LDY     #PLYARG/$100
ERROR UNDEFINED  ******************************@
  9569 DD4C 20B5DD            JSR     FSTOR               ;SAVE ARG
  9570 DD4F 20C6DD            JSR     FMOVE               ;ARG->FR1
  9571 DD52 AE0000            LDX     FPTR2
ERROR UNDEFINED  ***********************@
  9572 DD55 AC0100            LDY     FPTR2+1
ERROR UNDEFINED  *************************@
  9573 DD58 2093DD            JSR     FLD0R               ;COEF->FR0 (INIT SUM)
  9574 DD5B C6EF              DEC     PLYCNT
  9575 DD5D F033              BEQ     PLYOUT              ;DONE ?
  9576 DD5F 20DBDA    PLYEV1  JSR     FMUL                ; SUM * ARG
  9577 DD62 B02E              BCS     PLYOUT              ; O'FLOW
  9578 DD64 18                CLC
  9579 DD65 AD0000            LDA     FPTR2               ;BUMP COEF POINTER
ERROR UNDEFINED  ***********************@
  9580 DD68 6906              ADC     #FPREC
  9581 DD6A 8D0000            STA     FPTR2
ERROR UNDEFINED  ***********************@
  9582 DD6D 9008              BCC     PLYEV2
  9583 DD6F AD0100            LDA     FPTR2+1             ;ACROSS PAGE
ERROR UNDEFINED  *************************@
  9584 DD72 6900              ADC     #0
  9585 DD74 8D0100            STA     FPTR2+1
ERROR UNDEFINED  *************************@
  9586 DD77 AE0000    PLYEV2  LDX     FPTR2
ERROR UNDEFINED  ***********************@
  9587 DD7A AC0100            LDY     FPTR2+1
ERROR UNDEFINED  *************************@
  9588 DD7D 20A4DD            JSR     FLD1R               ;GET NEXT COEF
  9589 DD80 2066DA            JSR     FADD                ;SUM*ARG + COEF
  9590 DD83 B00D              BCS     PLYOUT              ; O'FLOW
  9591 DD85 C6EF              DEC     PLYCNT
  9592 DD87 F009              BEQ     PLYOUT              ;DONE ?
  9593 DD89 A200              LDX     #PLYARG&$FF
ERROR UNDEFINED  *****************************@
  9594 DD8B A000              LDY     #PLYARG/$100
ERROR UNDEFINED  ******************************@
  9595 DD8D 20A4DD            JSR     FLD1R               ;GET ARG AGAIN
  9596 DD90 30CD              BMI     PLYEV1              ; (=JMP)
```

```
 9590 DD83 B00D                 BCS     PLYOUT          ; O'FLOW
 9591 DD85 C6EF                 DEC     PLYCNT
 9592 DD87 F009                 BEQ     PLYOUT          ;DONE ?
 9593 DD89 A200                 LDX     #PLYARG&$FF
ERROR UNDEFINED   *****************************@
 9594 DD8B A000                 LDY     #PLYARG/$100
ERROR UNDEFINED   ******************************@
 9595 DD8D 20A4DD               JSR     FLD1R           ;GET ARG AGAIN
 9596 DD90 30CD                 BMI     PLYEV1          ;  (=JMP)
```

```
 9597 DD92 60         PLYOUT  RTS
 9598
```

```
                                 PAGE     'FLOATING LOAD/STORE'
 9599                    *       LOAD FR0 FROM (X,Y)  X=LSB, Y=MSB, USES FLPTR (PG0)
 9600 DD93 8E0000   FLD0R    STX      FLPTR              ; SET FLPTR => (X,Y)
ERROR UNDEFINED   ************************@
 9601 DD96 8C0100            STY      FLPTR+1
ERROR UNDEFINED   **************************@
 9602 DD99 A005     FLD0P    LDY      #FPREC-1           ;# BYTES ENTER HERE W/FLPTR SET
 9603 DD9B B100     FLD01    LDA      (FLPTR),Y          ; MOVE
ERROR UNDEFINED   *************************@
 9604 DD9D 99D400            STA      FR0,Y
 9605 DDA0 88                DEY
 9606 DDA1 10F8              BPL      FLD01              ; COUNT & LOOP
 9607 DDA3 60                RTS
 9608                    *
 9609                    *       LOAD FR1 FROM (X,Y) OR (FLPTR)
 9610 DDA4 8E0000   FLD1R    STX      FLPTR              ; FLPTR=>(X,Y)
ERROR UNDEFINED   ************************@
 9611 DDA7 8C0100            STY      FLPTR+1
ERROR UNDEFINED   **************************@
 9612 DDAA A005     FLD1P    LDY      #FPREC-1           ; # BYTES ENTER W/FLPTR SET
 9613 DDAC B100     FLD11    LDA      (FLPTR),Y          ; MOVE
ERROR UNDEFINED   *************************@
 9614 DDAE 99E000            STA      FR1,Y
 9615 DDB1 88                DEY
 9616 DDB2 10F8              BPL      FLD11              ; COUNT & LOOP
 9617 DDB4 60                RTS
 9618                    *
 9619                    *       STORE FR0 IN (X,Y) OR (FLPTR)
 9620 DDB5 8E0000   FST0R    STX      FLPTR
ERROR UNDEFINED   ************************@
 9621 DDB8 8C0100            STY      FLPTR+1
ERROR UNDEFINED   **************************@
 9622 DDBB A005     FST0P    LDY      #FPREC-1           ; ENTRY W/FLPTR SET
 9623 DDBD B9D400   FST01    LDA      FR0,Y
 9624 DDC0 9100              STA      (FLPTR),Y
ERROR UNDEFINED   *************************@
 9625 DDC2 88                DEY
 9626 DDC3 10F8              BPL      FST01
 9627 DDC5 60                RTS
 9628                    *
 9629                    *       MOVE FR0 TO FR1
 9630                    *
 9631                    MV0TO1
 9632 DDC6 A205     FMOVE    LDX      #FPREC-1
 9633 DDC8 B5D4     FMOVE1   LDA      FR0,X
 9634 DDCA 95E0              STA      FR1,X
 9635 DDCC CA                DEX
 9636 DDCD 10F9              BPL      FMOVE1
 9637 DDCF 60                RTS
 9638
```

```
                                  PAGE     'EXP(X) & EXP10(X)'
 9639 DDD0 A299         EXP       LDX      #LOG10E&$FF        ; E**X = 10**(X*LOG10(E))
 9640 DDD2 A0DE                   LDY      #LOG10E/$100
 9641 DDD4 20A4DD                 JSR      FLD1R
 9642 DDD7 20DBDA                 JSR      FMUL
 9643 DDDA B07F                   BCS      EXPERR
 9644 DDDC A900         EXP10     LDA      #0                 ; 10**X
 9645 DDDE 85F1                   STA      XFMFLG             ; CLEAR TRANSFORM FLAG
 9646 DDE0 A5D4                   LDA      FR0
 9647 DDE2 85F0                   STA      SGNFLG             ; REMEMBER ARG SGN
 9648 DDE4 297F                   AND      #$7F               ; ; & MAKE PLUS
 9649 DDE6 85D4                   STA      FR0
 9650 DDE8 38                     SEC
 9651 DDE9 E940                   SBC      #$40
 9652 DDEB 3026                   BMI      EXP1               ; X<1 SO USE SERIES DIRECTLY
 9653                   *         10**X = 10**(I+F) = (10**I) * (10**F)
 9654 DDED C904                   CMP      #FPREC-2
 9655 DDEF 106A                   BPL      EXPERR             ; ARG TOO BIG
 9656 DDF1 A200                   LDX      #FPSCR&$FF
ERROR UNDEFINED   ****************************@
 9657 DDF3 A000                   LDY      #FPSCR/$100
ERROR UNDEFINED   *****************************@
 9658 DDF5 20B5DD                 JSR      FSTOR              ; SAVE ARG
 9659 DDF8 20D2D9                 JSR      FPI                ; MAKE INTEGER
 9660 DDFB A5D4                   LDA      FR0
 9661 DDFD 85F1                   STA      XFMFLG             ; SAVE MULTIPLIER EXP IN XFORM
 9662 DDFF A5D5                   LDA      FR0+1              ; CHECK MSB
 9663 DE01 D058                   BNE      EXPERR             ; SHUDNT OTTA HAVE ANY
 9664 DE03 20AAD9                 JSR      IFP                ; NOW TURN IT BACK TO FLPT
 9665 DE06 20C6DD                 JSR      FMOVE
 9666 DE09 A200                   LDX      #FPSCR&$FF
ERROR UNDEFINED   ****************************@
 9667 DE0B A000                   LDY      #FPSCR/$100
ERROR UNDEFINED   *****************************@
 9668 DE0D 2093DD                 JSR      FLD0R              ; GET ARG BACK
 9669 DE10 2060DA                 JSR      FSUB               ; ARG - INTEGER PART = FRACTION PART
 9670                   *         NOW HAVE FRACTION PART OF ARG (F) IN FR0,
 9671                   *         INTEGER PART (I)
 9672                   *         IN XFMFLG.  USE SERIES APPROX FOR
 9673                   *         10**F, THEN MULTIPLY BY 10**I
 9674                   EXP1
 9675 DE13 A90A                   LDA      #NPCOEF
 9676 DE15 A25D                   LDX      #P10COF&$FF
 9677 DE17 A0DE                   LDY      #P10COF/$100
 9678 DE19 2040DD                 JSR      PLYEVL             ;P(X)
 9679 DE1C 20C6DD                 JSR      FMOVE
 9680 DE1F 20DBDA                 JSR      FMUL               ;P(X)*P(X)
 9681 DE22 A5F1                   LDA      XFMFLG             ; DID WE TRANSFORM ARG
 9682 DE24 F023                   BEQ      EXPSGN             ; NO SO LEAVE RESULT ALONE
 9683 DE26 18                     CLC
 9684 DE27 6A                     RORA                        ; I/2
 9685 DE28 85E0                   STA      FR1                ; SAVE AS EXP-TO-BE
 9686 DE2A A901                   LDA      #1                 ; GET MANTISSA BYTE
 9687 DE2C 9002                   BCC      EXP2               ; CHECK BIT SHIFTED OUT OF A UP THERE
 9688 DE2E A910                   LDA      #$10               ; I WAS ODD - MANTISSA = 10
 9689 DE30 85E1      EXP2    STA   FR1+1
 9690 DE32 A204              LDX   #FPREC-2
 9691 DE34 A900              LDA   #0
 9692 DE36 95E2      EXP3    STA   FR1+2,X          ; CLEAR REST OF MANTISSA
 9693 DE38 CA                DEX
 9694 DE39 10FB              BPL   EXP3
 9695 DE3B A5E0              LDA   FR1              ; BACK TO EXPONENT
 9696 DE3D 18                CLC
 9697 DE3E 6940              ADC   #$40             ; BIAS IT
 9698 DE40 B019              BCS   EXPERR           ; OOPS...IT'S TOO BIG
 9699 DE42 3017              BMI   EXPERR
 9700 DE44 85E0              STA   FR1              ; FR1 = 10**I
 9701 DE46 20DBDA            JSR   FMUL             ; (10**I)*(10**F)
 9702 DE49 A5F0      EXPSGN  LDA   SGNFLG           ; WAS ARG<0
 9703 DE4B 100D              BPL   EXPOUT           ; NO-DONE
 9704 DE4D 20C6DD            JSR   FMOVE            ; YES-INVERT RESULT
 9705 DE50 A29F              LDX   #FONE&$FF
 9706 DE52 A0DE              LDY   #FONE/$100
 9707 DE54 2093DD            JSR   FLDOR
 9708 DE57 2028DB            JSR   FDIV
 9709 DE5A 60        EXPOUT  RTS                    ; (PANT, PANT - FINISHED!!)
 9710 DE5B 38        EXPERR  SEC                    ; FLAG ERROR
 9711 DE5C 60                RTS                    ; & QUIT
 9712 DE5D 3D        P10COF  .BYTE $3D,$17,$94,$19,0,0 ;0.0000179419
      DE5E 17
      DE5F 94
      DE60 19
      DE61 00
      DE62 00
 9713 DE63 3D                .BYTE $3D,$57,$33,$05,0,0 ;0.0000573305
      DE64 57
      DE65 33
      DE66 05
      DE67 00
      DE68 00
 9714 DE69 3E                .BYTE $3E,$05,$54,$76,$62,0 ;0.0005547662
      DE6A 05
      DE6B 54
      DE6C 76
      DE6D 62
      DE6E 00
 9715 DE6F 3E                .BYTE $3E,$32,$19,$62,$27,0 ;0.0032176227
      DE70 32
      DE71 19
      DE72 62
      DE73 27
      DE74 00
 9716 DE75 3F                .BYTE $3F,$01,$68,$60,$30,$36 ;0.0168603036
      DE76 01
      DE77 68
      DE78 60
      DE79 30
      DE7A 36
 9717 DE7B 3F                .BYTE $3F,$07,$32,$03,$27,$41 ;0.0732032741
      DE7C 07
      DE7D 32
      DE7E 03
      DE7F 27
      DE80 41
 9718 DE81 3F                   BYTE   $3F,$25,$43,$34,$56,$75 ;0.2543345675
      DE82 25
      DE83 43
      DE84 34
      DE85 56
      DE86 75
 9719 DE87 3F                   BYTE   $3F,$66,$27,$37,$30,$50 ;0.6627373050
      DE88 66
      DE89 27
      DE8A 37
      DE8B 30
      DE8C 50
 9720 DE8D 40                   BYTE   $40,$01,$15,$12,$92,$55 ;1.15129255
      DE8E 01
      DE8F 15
      DE90 12
      DE91 92
      DE92 55
 9721 DE93 3F                   BYTE   $3F,$99,$99,$99,$99,$99 ;0.999999999
      DE94 99
      DE95 99
      DE96 99
      DE97 99
      DE98 99
 9722      000A      NPCOEF  EQU    *-P10COF/FPREC
 9723 DE99 3F        LOG10E  BYTE   $3F,$43,$42,$94,$48,$19  ; LOG10(E)
      DE9A 43
      DE9B 42
      DE9C 94
      DE9D 48
      DE9E 19
 9724 DE9F 40        FONE    BYTE   $40,1,0,0,0,0   ; 1.0
      DEA0 01
      DEA1 00
      DEA2 00
      DEA3 00
      DEA4 00
 9725
```

```
                             PAGE    'Z = (X-C)/(X+C)'
 9726 DEA5 9E0000    XFORM   STX     FPTR2
ERROR UNDEFINED   ***********************@
 9727 DEA8 8C0100            STY     FPTR2+1
ERROR UNDEFINED   *************************@
 9728 DEAB A200              LDX     #PLYARG&$FF
ERROR UNDEFINED   *****************************@
 9729 DEAD A000              LDY     #PLYARG/$100
ERROR UNDEFINED   ******************************@
 9730 DEAF 20B5DD            JSR     FSTOR            ; STASH X IN PLYARG
 9731 DEB2 AE0000            LDX     FPTR2
ERROR UNDEFINED   ***********************@
 9732 DEB5 AC0100            LDY     FPTR2+1
ERROR UNDEFINED   *************************@
 9733 DEB8 20A4DD            JSR     FLD1R
 9734 DEBB 2066DA            JSR     FADD             ; X+C
 9735 DEBE A200              LDX     #FPSCR&$FF
ERROR UNDEFINED   ****************************@
 9736 DEC0 A000              LDY     #FPSCR/$100
ERROR UNDEFINED   *****************************@
 9737 DEC2 20B5DD            JSR     FSTOR
 9738 DEC5 A200              LDX     #PLYARG&$FF
ERROR UNDEFINED   *****************************@
 9739 DEC7 A000              LDY     #PLYARG/$100
ERROR UNDEFINED   ******************************@
 9740 DEC9 2093DD            JSR     FLD0R
 9741 DECC AE0000            LDX     FPTR2
ERROR UNDEFINED   ***********************@
 9742 DECF AC0100            LDY     FPTR2+1
ERROR UNDEFINED   *************************@
 9743 DED2 20A4DD            JSR     FLD1R
 9744 DED5 2060DA            JSR     FSUB             ; X-C
 9745 DED8 A200              LDX     #FPSCR&$FF
ERROR UNDEFINED   ****************************@
 9746 DEDA A000              LDY     #FPSCR/$100
ERROR UNDEFINED   *****************************@
 9747 DEDC 20A4DD            JSR     FLD1R
 9748 DEDF 2028DB            JSR     FDIV             ; (X-C)/(X+C) = Z
 9749 DEE2 60                RTS
 9750
```

```
                                PAGE    'LOG10(X)'
 9751 DEE3 A901      LOG        LDA     #1                 ;REMEMBER ENTRY POINT
 9752 DEE5 D002                 BNE     LOGBTH
 9753 DEE7 A900      LOG10      LDA     #0                 ;  CLEAR FLAG
 9754 DEE9 85F0      LOGBTH     STA     SGNFLG             ;  USE SGNFLG FOR LOG/LOG10 MARKER
 9755 DEEB A5D4                 LDA     FR0
 9756 DEED 1002                 BPL     LOG5
 9757 DEEF 38        LOGERR     SEC
 9758 DEF0 60                   RTS
 9759                LOG5
 9760                *          WE WANT X = F*(10**Y), 1<F<10
 9761                *          10**Y HAS SAME EXP BYTE AS X
 9762                *          & MANTISSA BYTE = 1 OR 10
 9763 DEF1 A5D4      LOG1       LDA     FR0
 9764 DEF3 85E0                 STA     FR1
 9765 DEF5 38                   SEC
 9766 DEF6 E940                 SBC     #$40
 9767 DEF8 0A                   ASLA
 9768 DEF9 85F1                 STA     XFMFLG             ;  REMEMBER Y
 9769 DEFB A5D5                 LDA     FR0+1
 9770 DEFD 29F0                 AND     #$F0
 9771 DEFF D004                 BNE     LOG2
 9772 DF01 A901                 LDA     #1
 9773 DF03 D004                 BNE     LOG3
 9774 DF05 E6F1      LOG2       INC     XFMFLG             ;  BUMP Y
 9775 DF07 A910                 LDA     #$10
 9776 DF09 85E1      LOG3       STA     FR1+1              ;  SET UP MANTISSA
 9777 DF0B A204                 LDX     #FPREC-2           ;  CLEAR REST OF MANTISSA
 9778 DF0D A900                 LDA     #0
 9779 DF0F 95E2      LOG4       STA     FR1+2,X
 9780 DF11 CA                   DEX
 9781 DF12 10FB                 BPL     LOG4
 9782 DF14 2028DB               JSR     FDIV               ;  X = X/(10**Y) - S.B.  IN (1,10)
 9783                FLOG10                                ;;LOG10(X),1<=X<=10
 9784 DF17 A27C                 LDX     #SQR10&$FF
 9785 DF19 A0DF                 LDY     #SQR10/$100
 9786 DF1B 20A5DE               JSR     XFORM              ;Z = (X-C)/(X+C),C*C = 10
 9787 DF1E A200                 LDX     #FPSCR&$FF
ERROR UNDEFINED     ******************************@
 9788 DF20 A000                 LDY     #FPSCR/$100
ERROR UNDEFINED     ******************************@
 9789 DF22 20B5DD               JSR     FSTOR              ;SAVE Z
 9790 DF25 20C6DD               JSR     FMOVE
 9791 DF28 20DBDA               JSR     FMUL               ;  Z*Z
 9792 DF2B A90A                 LDA     #NLCOEF
 9793 DF2D A288                 LDX     #LGCOEF&$FF
 9794 DF2F A0DF                 LDY     #LGCOEF/$100
 9795 DF31 2040DD               JSR     PLYEVL             ;  P(Z*Z)
 9796 DF34 A200                 LDX     #FPSCR&$FF
ERROR UNDEFINED     ******************************@
 9797 DF36 A000                 LDY     #FPSCR/$100
ERROR UNDEFINED     ******************************@
 9798 DF38 20A4DD               JSR     FLD1R
 9799 DF3B 20DBDA               JSR     FMUL               ;  Z*P(Z*Z)
 9800 DF3E A282                 LDX     #FHALF&$FF
```

```
9801 DF40 A0DF                  LDY    #FHALF/$100
9802 DF42 20A4DD                JSR    FLD1R
9803 DF45 2066DA                JSR    FADD              ; 0.5 + Z*P(Z*Z)
9804 DF48 20C6DD                JSR    FMOVE
9805 DF4B A900                  LDA    #0
9806 DF4D 85D5                  STA    FR0+1
9807 DF4F A5F1                  LDA    XFMFLG
9808 DF51 85D4                  STA    FR0
9809 DF53 1007                  BPL    LOG6
9810 DF55 49FF                  EOR    #-1               ; FLIP SIGN
9811 DF57 18                    CLC
9812 DF58 6901                  ADC    #1
9813 DF5A 85D4                  STA    FR0
9814                  LOG6
9815 DF5C 20AAD9                JSR    IFP               ; LEAVES FR1 ALONE
9816 DF5F 24F1                  BIT    XFMFLG
9817 DF61 1006                  BPL    LOG7
9818 DF63 A980                  LDA    #$80              ; FLIP AGAIN
9819 DF65 05D4                  ORA    FR0
9820 DF67 85D4                  STA    FR0
9821                  LOG7
9822 DF69 2066DA                JSR    FADD              ; LOG(X) = LOG(X) +Y
9823                  LOGOUT
9824 DF6C A5F0                  LDA    SGNFLG
9825 DF6E F00A                  BEQ    LOGDON            ;WAS LOG10, NOT LOG
9826 DF70 A299                  LDX    #LOG10E&255       ; LOG(X)/LOG10(E)
9827 DF72 A0DE                  LDY    #LOG10E/$100
9828 DF74 20A4DD                JSR    FLD1R
9829 DF77 2028DB                JSR    FDIV
9830 DF7A 18          LOGDON    CLC
9831 DF7B 60                    RTS
9832 DF7C 40          SQR10     .BYTE  $40,$03,$16,$22,$77,$66 ;SQUARE ROOT OF 10
     DF7D 03
     DF7E 16
     DF7F 22
     DF80 77
     DF81 66
9833 DF82 3F          FHALF     .BYTE  $3F,$50,0,0,0,0          ; 0.5
     DF83 50
     DF84 00
     DF85 00
     DF86 00
     DF87 00
9834 DF88 3F          LGCOEF    .BYTE  $3F,$49,$15,$57,$11,$08 ;0.4915571108
     DF89 49
     DF8A 15
     DF8B 57
     DF8C 11
     DF8D 08
9835 DF8E BF                    .BYTE  $BF,$51,$70,$49,$47,$08 ;-0.5170494708
     DF8F 51
     DF90 70
     DF91 49
     DF92 47
     DF93 08
```

```
 9836 DF94 3F                     .BYTE   $3F,$39,$20,$57,$61,$95 ;0.3920576195
      DF95 39
      DF96 20
      DF97 57
      DF98 61
      DF99 95
 9837 DF9A BF                     .BYTE   $BF,$04,$39,$63,$03,$55 -0.0439630355
      DF9B 04
      DF9C 39
      DF9D 63
      DF9E 03
      DF9F 55
 9838 DFA0 3F                     .BYTE   $3F,$10,$09,$30,$12,$64 0.1009301264
      DFA1 10
      DFA2 09
      DFA3 30
      DFA4 12
      DFA5 64
 9839 DFA6 3F                     .BYTE   $3F,$09,$39,$08,$04,$60   ; 0.0939080460
      DFA7 09
      DFA8 39
      DFA9 08
      DFAA 04
      DFAB 60
 9840 DFAC 3F                     .BYTE   $3F,$12,$42,$58,$47,$42 ;0.1242584742
      DFAD 12
      DFAE 42
      DFAF 58
      DFB0 47
      DFB1 42
 9841 DFB2 3F                     .BYTE   $3F,$17,$37,$12,$06,$08   ; 0.1737120608
      DFB3 17
      DFB4 37
      DFB5 12
      DFB6 06
      DFB7 08
 9842 DFB8 3F                     .BYTE   $3F,$28,$95,$29,$71,$17 ;0.28957117
      DFB9 28
      DFBA 95
      DFBB 29
      DFBC 71
      DFBD 17
 9843 DFBE 3F                     .BYTE   $3F,$86,$85,$88,$96,$44 ;0.8685889644
      DFBF 86
      DFC0 85
      DFC1 88
      DFC2 96
      DFC3 44
 9844           000A     NLCOEF   EQU     *-LGCOEF/FPREC
 9845 DFC4 3E            ATCOEF   .BYTE   $3E,$16,$05,$44,$49,0 ,0.0016054449
      DFC5 16
      DFC6 05
      DFC7 44
      DFC8 49
      DFC9 00
 9846 DFCA BE               .BYTE   $BE,$95,$68,$38,$45,0 ;-0.009568345
      DFCB 95
      DFCC 68
      DFCD 38
      DFCE 45
      DFCF 00
 9847 DFD0 3F               .BYTE   $3F,$02,$68,$79,$94,$16 ;0.0268799416
      DFD1 02
      DFD2 68
      DFD3 79
      DFD4 94
      DFD5 16
 9848 DFD6 BF               .BYTE   $BF,$04,$92,$78,$90,$80 ;-0.0492789080
      DFD7 04
      DFD8 92
      DFD9 78
      DFDA 90
      DFDB 80
 9849 DFDC 3F               .BYTE   $3F,$07,$03,$15,$20,0 ;0.0703152000
      DFDD 07
      DFDE 03
      DFDF 15
      DFE0 20
      DFE1 00
 9850 DFE2 BF               .BYTE   $BF,$08,$92,$29,$12,$44 ;-0.0892291244
      DFE3 08
      DFE4 92
      DFE5 29
      DFE6 12
      DFE7 44
 9851 DFE8 3F               .BYTE   $3F,$11,$08,$40,$09,$11 ;0.1108400911
      DFE9 11
      DFEA 08
      DFEB 40
      DFEC 09
      DFED 11
 9852 DFEE BF               .BYTE   $BF,$14,$28,$31,$56,$04 ;-0.1428315604
      DFEF 14
      DFF0 28
      DFF1 31
      DFF2 56
      DFF3 04
 9853 DFF4 3F               .BYTE   $3F,$19,$99,$98,$77,$44 0.1999987744
      DFF5 19
      DFF6 99
      DFF7 98
      DFF8 77
      DFF9 44
 9854 DFFA BF               .BYTE   $BF,$33,$33,$33,$31,$13  ; -0.3333333113
      DFFB 33
      DFFC 33
      DFFD 33
      DFFE 31
      DFFF 13
 9855 E000 3F          FP9S .BYTE   $3F,$99,$99,$99,$99,$99  ; 0.999999999
```

```
      DFFE 31
      DFFF 13
 9855 E000 3F          FP9S    .BYTE   $3F,$99,$99,$99,$99,$99  ; 0.999999999
```

```
      E001 99
      E002 99
      E003 99
      E004 99
      E005 99
 9856      000B        NATCF   EQU     *-ATCOEF/FPREC
 9857 E006 3F          PIOV4   .BYTE   $3F,$78,$53,$98,$16,$34  ; PI/4 = ARCTAN(1.0)
      E007 78
      E008 53
      E009 98
      E00A 16
      E00B 34
 9858
```

```
                                      PAGE     'ATARI CARTRIDGE VECTORS'
 9859       0001                      .IF      ATARIV
 9860       E00C                      ORG      CRTGI
 9861                       SCVECT
ERROR DUP   DEF        *********@
 9862 BFF9  60                        RTS
 9863 BFFA  00A0                      DB       @@COLDSTART        ;  COLDSTART ADDR
 9864 BFFC  00                        DB       0                  ;  CART EXISTS
 9865 BFFD  05                        DB       5                  ;  FLAG
 9866 BFFE  F9BF                      DB       @@SCVECT           ;  COLDSTART ENTRY ADDR
 9867                                 .ENDIF
 9868
 9869       0000                  PAGE     'END OF BASIC'
ASECT PTRS                         END

TSECT PTRS

BSECT PTRS


SYMBOL MAP

ABORT   BB70  A      ABRK    B58E  A      ABS     0002  A      ABSX    0080  A      ABSY    0001  A
ACU     0008  A      ADD     BAD9  A      ADDRON  0000  A      ADISPL  0492  A      ADRBIT  B9F4  A
ADSTBL  A7AE  A      ADXTBL  A963  A      AFP     D800  A      AINIT   BA1C  A      AINIT2  BA32  A
ALIST   BBCB  A*     ALIST1  BBDD  A      AMODE   FFFF  A      AND     BACC  A      APDBV   0000  A
APHM    000E  A      AQ1FLG  0099  A      ARES1   BA7A  A      ARES2   BA88  A      ARESET  BA7E  A
ASCIN   D800  A*     ASDIR   A198  A      ASIMP   A189  A      ASM1A   B59C  A      ASM1B   B67B  A
ASM1C   B687  A      ASM1D   B686  A      ASM1L   B597  A      ASM3    B56A  A      ASMISC  A1B2  A
ASMR    B591  A*     ASMULT  A17C  A      ASREL   A1A8  A      ASTBL   A7C4  A      ASYNTA  A0C6  A
ATARIV  0001  A      ATCOEF  DFC4  A      ATR1    0000  A*     AUTO    BCC6  A      AUTOFF  0000  A
AUTOFL  00A6  A      AUTON   0080  A      AXCOM   B5C6  A      AXDIR   B5E6  A      AXIMP   B8BF  A
AXMISC  B8F8  A      AXMULT  B916  A      AXREL   B8C8  A      AXTBL   A957  A      BELL    00FD  A
BININT  00D4  A      BIT8    BB57  A      BLKCNT  00FD  A      BRKBYT  0011  A      BRKINI  AFA5  A*
BRKPAT  B165  A*     BRKVEC  0206  A      BUG1    AAB1  A      BUGMSG  A07B  A      BYELOC  E471  A
C       0080  A      CA      0061  A      CAND    0071  A      CCOM    007A  A      CDELIM  007B  A
CDISPL  0493  A      CDIV    0070  A      CDO     0059  A      CDOL    0076  A      CDQ     0072  A
CDUM    0080  A      CEASM   004B  A*     CEBUG   006C  A      CEDIR   0013  A      CEEDIT  0060  A
CEIMP   002C  A      CEJECT  004F  A      CEMISC  0036  A      CEMNEM  004B  A*     CEMULT  004B  A
CEOL    0079  A      CEOP    007E  A*     CEOPT   0050  A*     CEREL   0034  A      CERR    004E  A
CFILE   0078  A      CIFDEL  007E  A      CIMM    0078  A      CIO     E456  A      CIX     00F2  A
CJMP    0036  A      CJSR    0035  A      CKREP   ACAF  A      CLALL1  BE4D  A      CLALL7  BE4A  A
CLASF   A5C7  A      CLFILE  BE27  A      CLIST   0050  A      CLOC    006F  A      CLPRN   0074  A
CLR1    A2A2  A      CLRMEM  A27C  A      CLSALL  BE48  A      CLSYSD  BE37  A      CM      0062  A
CMINUS  006E  A      CMUL    006F  A      CNO     004C  A      CNODO   009B  A      CNTON   0080  A
COBJ    004D  A      COLDST  A000  A      COMEXE  A0B4  A      COMSYN  A080  A      CONTLO  AA06  A
CONTRA  AA08  A*     CONTXT  AA03  A      COPTR   00A2  A      COX     0092  A      CPC     0097  A
CPLUS   006D  A      CQ      007D  A      CR      009B  A      CREL    B3B2  A      CRPRN   0075  A
CRTGI   BFF9  A      CSADR   0095  A      CSASM   0009  A*     CSBUG   0061  A*     CSC     0077  A*
CSDIR   0009  A      CSEDIT  0050  A      CSIMP   0014  A*     CSMISC  0035  A*     CSMNEM  0014  A
CSMULT  0037  A      CSOP    006D  A*     CSOPT   004C  A*     CSQ     0073  A      CSREL   002D  A*
CURLN   00A2  A      CVAFP   D800  A      CVFASC  D8E6  A*     CVIFP   D9AA  A*     CX      006B  A
CY      007C  A      DBVER   0000  A      DBYTEO  0080  A      DCBORG  0300  A*     DEADR   00A2  A
DEGFLG  00FB  A*     DELC    AB2D  A      DELIM   003C  A      DELLN   A237  A      DIGRT   00F1  A
DIMP    B3E2  A      DISASM  B297  A      DISFLG  00D2  A      DIV     BAF5  A      DMISC   B3EC  A
DMODE   0001  A      DMULTI  B3F1  A      DOORG   B7B2  A      DOREP   ACD6  A      DOSLOC  000A  A
DREL    B3E7  A      DSADR   009A  A      DSPFLG  02FE  A      DSTBL   A796  A*     DSYN2   A056  A
DSYNTA  A060  A      DXTBL   A93F  A*     ECOM    A035  A      ECSIZE  009E  A      EDERR   BEC9  A
EDMSG   A052  A      EEXP    00ED  A      EMODE   0000  A      ENTDTD  00A9  A      EOFFLG  0484  A
EOP1    0023  A      EOP2    0015  A      EQULBL  0002  A      ERADR   000C  A      ERASYN  0006  A
ERDUP   0007  A      EREQU   0009  A      ERGARB  0010  A      ERLBIG  000F  A      ERLMEM  0012  A
ERLNUM  0011  A      ERLOAD  0004  A      ERMSYN  0003  A      ERNDEF  0005  A      ERNLN   0002  A
ERNORG  0013  A      ERNSTR  000B  A      EROVFL  0014  A      EROVR   0008  A      ERPHAS  000D  A
ERRBUF  0487  A      ERRDEX  00A8  A      ERRMSG  BF2C  A      ERRNUM  00A1  A      ERROR   BEC9  A
ERUFR   000E  A      ERVAL   000A  A      ESIGN   00EF  A      ESTBL   A774  A      ESYN2   A016  A
ESYNTA  A025  A      ETXT    A1C2  A      EXBUFF  0082  A      EXINIT  B10A  A      EXP     DDD0  A*
```

```
PAGE   306    SHEP  ATARI EDITOR
END OF BASIC

EXP1    DE13  A    EXP10   DDDC  A*   EXP2    DE30  A    EXP3    DE36  A    EXPAND  A981  A*
EXPERR  DE3B  A    EXPLOW  A97F  A    EXPOUT  DE5A  A    EXPSGN  DE49  A    EXPTXT  A979  A
EXTBL   A91D  A    FADD    DA66  A    FADR    B203  A    FAGET   B21B  A    FAGET2  B230  A
FASC    DBE6  A    FCHRFL  00F0  A    FDIV    DB2B  A    FHALF   DF82  A    FILEON  0080  A*
FINDC   AC87  A    FINDON  0000  A    FLDO1   DD9B  A    FLDOP   DD99  A*   FLDOR   DD93  A
FLD11   DDAC  A    FLD1P   DDAA  A*   FLD1R   DDA4  A    FLIST   BD74  A    FLOAD   BD7F  A
FLOAD2  BD7D  A    FLOG10  DF17  A*   FLPTR   0000  U    FMOVE   DDC6  A    FMOVE1  DDC8  A
FMOVER  AA52  A*   FMPREC  0005  A    FMUL    DADB  A    FOBJ    BD88  A    FONE    DE9F  A
FP9S    E000  A*   FPI     D9D2  A    FPORG   D800  A    FPREC   0006  A    FPSCR   0000  U
FPTR2   0000  U    FRO     00D4  A    FROEX   00D6  A    FROM    00D5  A    FR1     00E0  A
FR1M    00E1  A    FR2     00E6  A    FRA10   DD01  A    FRA1E   DD09  A    FRA20   DD05  A
FRA2E   DD0F  A    FRE     00DA  A    FRINIT  AD6D  A    FRSIGN  00EE  A    FRX     00EC  A
FSTO1   DDBD  A    FSTOP   DDBB  A*   FSTOR   DDB5  A    FSUB    DA60  A    GASML   B65A  A
GCXI    BD5D  A    GETC1   AC5A  A    GETC2   AC58  A    GETC3   AC56  A    GETCOD  AC5C  A
GETHEX  A45E  A    GETIN   AC63  A    GETLL   BEB5  A    GETSTM  BE8A  A    GJADR   B135  A
GJADR1  B137  A    GLGO    BD2C  A    GLPCX   BD61  A    GLPCX1  BD64  A*   GLPX    BD64  A
GNLINE  BD28  A    GNXTL   BEBA  A    GNXTL1  BEBB  A    GOBJ    AE2C  A    GPCHAR  B23A  A
HEXTAB  BF6C  A    HIMEM   02E5  A    HMADR   02E5  A    IASC    BF4E  A    ICAUX1  034A  A
ICAUX2  034B  A    ICAUX3  034C  A    ICAUX4  034D  A    ICAUX5  034E  A    ICBAH   0345  A
ICBAL   0344  A    ICBLH   0349  A    ICBLL   0348  A    ICCLOS  000C  A    ICCOM   0342  A
ICDDC   000E  A*   ICDNO   0341  A*   ICFREE  00FF  A*   ICGBC   0006  A    ICGBR   0004  A*
ICGTC   0007  A*   ICGTR   0005  A    ICHID   0340  A*   ICLEN   0010  A    ICMAX   000E  A*
ICOIN   0001  A*   ICOIO   0003  A    ICOOUT  0002  A*   ICPBC   000A  A    ICPBR   0008  A*
ICPTC   000B  A*   ICPTR   0009  A*   ICPUT   0346  A    ICSBRK  0080  A    ICSDER  0083  A*
ICSDNR  0081  A*   ICSEOF  0003  A*   ICSIVC  0084  A*   ICSIVN  0086  A*   ICSNED  0082  A*
ICSNOP  0085  A*   ICSOK   0001  A*   ICSTA   0343  A    ICSTAT  000D  A*   ICSTR   0002  A*
ICSWPE  0087  A*   IDISPL  0491  A    IFP     D9AA  A    ILEN    0481  A    ILSHFT  DA5A  A
IMM     0001  A    INBOUT  ABEB  A    INBUFF  00F3  A    INCLA   B3A3  A    INCMVF  B233  A
INCPST  AD95  A    INCSRC  A625  A    INDX    0010  A    INDY    0020  A    INTLBF  DA51  A
INVOPT  B452  A    IO1     BE56  A    IO2     BE5A  A*   IO3     BE5C  A    IO4     BE5E  A*
IO5     BE65  A*   IO6     BE69  A*   IO7     BE70  A    IO8     BE73  A    IOCB    0340  A
IOCBOR  0340  A    IOCMD   0485  A    IODVC   0486  A    IOTES2  BDBE  A    IOTEST  BDBB  A
ITTAB   B3DA  A    KWSRCH  A5AE  A    KWSTRT  0008  A    KWTAB   A62F  A    KWTAB2  A65C  A
L       B745  A*   LBLS1   A550  A    LBLSRC  A524  A    LBPR1   057E  A*   LBPR2   057F  A*
LBUFF   0580  A    LCHK1   0084  A    LCHK2   0009  A    LDDVX   BDB2  A    LDIOST  BE41  A
LEADR   0097  A    LELNUM  00A2  A    LENTAB  BA12  A    LGCOEF  DF88  A    LINDEX  00D2  A
LINLNG  00A1  A    LISDV   0001  A    LIST    AB55  A    LIST1   ABA2  A    LISTC   AB81  A
LISTDT  00AA  A    LMADR   02E7  A    LMSAV   048A  A    LNCNT   0483  A    LNINCR  00AB  A
LNUMOF  0080  A    LNUMON  0000  A    LOADDV  0003  A    LOCCNT  00AD  A    LOG     DEE3  A*
LOG1    DEF1  A*   LOG10   DEE7  A*   LOG10E  DE99  A    LOG2    DF05  A    LOG3    DF09  A
LOG4    DF0F  A    LOG5    DEF1  A    LOG6    DF5C  A    LOG7    DF69  A    LOGBTH  DEE9  A
LOGDON  DF7A  A    LOGERR  DEEF  A*   LOGOUT  DF6C  A*   LOMEM   0080  A    LOMFLG  00A4  A
LONE    0040  A    LRANG1  AC14  A    LRANGE  AC17  A    LSADR   009C  A    LSEARC  AD3E  A
LSYNER  B5C0  A    LTHREE  00C0  A    LTWO    0080  A    LTYPFL  00D1  A    MAMODE  0080  A
MASYNT  B4B0  A    MAXCIX  0099  A    MDEND   DB1A  A    MDESUP  DCCF  A    MDSUP   DCE0  A
MEMFUL  0001  A    MEMTOP  0090  A    MEND1   DB1E  A*   MEND2   DB21  A    MEND3   DB24  A
MISCRA  0500  A    MODBYT  BA08  A    MODEFL  00A7  A    MOVLIN  ABA8  A    MOVNUM  BB3D  A
MOVTXT  ABD1  A    MSGADR  0093  A    MSGXY   BF9A  A    MSGXYC  BFB2  A    MUL     BB16  A
MVOTO1  DDC6  A*   MVFA    00E6  A    MVFEA   009E  A    MVFROE  DD34  A    MVFR12  DD28  A
MVLNG   009C  A    MVTA    00E8  A    NATCF   000B  A*   NIBSHO  DBEB  A    NIBSH2  DBE7  A
NLCOEF  000A  A    NOCDO   BE47  A    NOINV   0010  A    NORM    DC00  A    NORM1   DC04  A
NOROP   B047  A    NOROP1  B048  A    NPCOEF  000A  A    NSIGN   00EE  A    NTCMD   0026  A
NULFLG  00AF  A    NZ      0001  A    OBJBUF  0082  A    OBJDEX  0099  A    OBJDTD  00D3  A
OBJDV   0002  A    OBJEND  0084  A    OBJM2   BB65  A    OBJMAN  BB69  A    OCNTL   BB7C  A
OERR    0020  A    OFF     B899  A    OLBCNT  00A1  A    OLIST   0080  A    ONEBYT  0001  A
```

```
PAGE    307     SHEP   ATARI EDITOR
END OF BASIC


ODBJ      0040   A         OPAGE     0001   A         OPATYP    B9CA   A         OPEN      BD8E   A         OPSRCH    B463   A
OPSYN     A164   A         OPTFLG    0494   A         OPTTAB    B8BB   A         ORRES     BB48   A         OSRCH     B80B   A
OUT       B5B9   A         OUTBUF    0080   A         OUTCNT    00FD   A         P10COF    DE5D   A         PABS      00C8   A
PABSX     00CC   A         PABSY     00C9   A         PACU      0040   A         PAGE1     B6DF   A         PAGEND    0090   A
PAGEP     008E   A         PASS1     0080   A         PASS2     0040   A         PASSFL    00A5   A         PEJECT    000C   A
PIMM      00A0   A         PIN       BC28   A         PINDX     0096   A         PINDY     0093   A         PIOV4     E006   A*
PLST      000C   A         PLYARG    0000   U         PLYCNT    00EF   A         PLYEV1    DD5F   A         PLYEV2    DD77   A
PLYEVL    DD40   A         PLYOUT    DD92   A         PMASK     0482   A         PMTAB     B486   A         PNUM      AAC1   A
POBJL     BC01   A         PORG      B7F4   A         POX       0495   A         PRA       B35A   A         PRADR     B37E   A
PRADRB    BF86   A         PRBLAN    BC83   A         PRBLK1    BF7C   A         PRBLK3    BC81   A         PRCHAR    BD35   A
PRCNT     00FC   A         PRCR      BF81   A         PRCX      BD37   A*        PRHEX2    BF54   A         PRHEXB    BF8C   A
PRHEXC    BF92   A         PRIHEX    BF5A   A         PRINUM    BF37   A         PRPAD     BC76   A         PRPAGE    B70C   A
PRPC      B398   A         PRPL      BC62   A         PRREL     B365   A         PRTERR    BEF6   A         PRTITL    B706   A
PTBUFF    0093   A         PTCMD     0025   A         PTDEX     00D0   A         PTSTR     B693   A         PUTC1     BD37   A
PUTCHA    BD35   A*        PUTO2     BD6A   A         PUTOBJ    BD70   A         PZPG      0088   A         PZPGX     008C   A
PZPGY     0089   A*        QTEMP     00D9   A         QUERRY    ACB3   A*        RADFLG    00FB   A*        RAM2      0480   A
REPFLG    00D3   A         REPON     0080   A         RESBYT    00D1   A         ROM       A000   A         RSHFOE    DC62   A
RSHFTO    DC3A   A         RSHFT1    DC3E   A         RSTEP     B06B   A         RSTEPO    B079   A         RSTEP1    B085   A
RSTEP2    B07D   A         RSTVB     B18E   A         RTNADR    00A2   A         SCRDV     0000   A         SCVECT    BFF9   A
SDEND     AF03   A         SEARCH    A5F2   A         SET1      0000   A         SETA      0001   A         SETBVB    E45C   A
SETQ      FFFF   A         SETUPC    B123   A         SGNFLG    00F0   A         SICKIO    BDC1   A         SIX       0500   A
SKBLAN    DBA1   A         SKCTL     020F   A         SKPBLA    DBA1   A         SMBLD     008C   A         SMBLEN    008A   A
SMBLTP    0088   A         SOX       0501   A         SPC       0502   A         SPTAB     B142   A         SQR10     DF7C   A
SQRCNT    00EF   A*        SRCADR    0093   A         SRCCNT    00A1   A         SRCNXT    A61C   A         SRCSKP    00A1   A
STACK     0500   A         STENUM    00D1   A         STEP      B00D   A         STKLVL    00A0   A         STMCUR    0086   A
STMSTR    00D2   A         STOP      00D0   A         STR1      B47A   A         STR1LN    00A0   A         STR2      B480   A
STR2LN    0491   A         STRDEL    00D1   A         STRDEX    0481   A         STRMOV    ABFF   A         SUB       BAE7   A
SVCIX     00FC   A         SVESA     0095   A         SVLENG    00FF   A         SYNINI    A264   A         TAB1      0480   A
TAB2      0481   A         TAB3      0482   A         TABCNT    00D0   A         TABEXP    BAC2   A         TABS      0007   A
TABSX     0009   A         TABSY     0008   A         TACU      0003   A         TADR      B1F6   A         TCOMM     0000   A
TDIR      0001   A         TEMPRL    048C   A         TEOE      0004   A         THEX2     0001   A         THRBYT    0003   A
TIMM      0002   A         TIMP      0002   A*        TINDX     0000   A         TINDY     0001   A         TITLEN    008E   A
TITLEP    008C   A         TLBL      0001   A         TLEND     AE6F   A         TMACRO    0006   A*        TMISC     0004   A*
TMULTI    0005   A         TNLBL     0000   A         TOBJDE    00FE   A         TOPT2     A7F3   A         TOPTBL    A7D0   A
TPCIX     00D0   A         TPVB1     AFCA   A         TPVBLK    AFC3   A         TREL      0003   A*        TSCOX     00D0   A
TSLNUM    009A   A         TSOE      0003   A*        TSTALP    A3B6   A         TSTBRK    BE7E   A         TSTEJT    BCA9   A
TSTLIS    BC8D   A         TSTLST    AE85   A         TSTNUM    DBAF   A         TSTOER    BBF6   A         TSTR      0002   A
TSTTEN    AE78   A         TVSCIX    00D0   A         TWOBYT    0002   A         TXTDV     0003   A*        TXTEND    0088   A
TXTTAB    0084   A         TYPEN     00F7   A         TYPEN2    00CE   A         TZPG      0004   A*        TZPGX     0006   A*
TZPGY     0005   A         UPAUTO    BD13   A         UPC       00AB   A         UPCADR    AF8D   A         UPCUR     BD13   A
URA       048C   A         UREG      048C   A         URP       048F   A         URS       0490   A         URX       048D   A
URY       048E   A         VBRK      B162   A         VERROR    BEC7   A         WAITCR    AB40   A         WARMFL    0008   A
WARMST    A015   A         WHAT      A0A3   A         WHTMSG    A0AF   A         WORDFL    00A1   A         WORDON    0000   A
XA        B4A5   A         XASM      B506   A         XATBUG    AA72   A         XBR       B0DB   A         XBR1      B0FB   A
XBRK      B0D8   A         XBYE      AA6C   A         XBYTE     B736   A         XC        AF55   A         XCLR      A288   A
XCR       AF4B   A         XD        AE99   A         XDBYTE    B717   A         XDEL      AB23   A         XDOS      AA6F   A
XDR       AF11   A         XEND      B5F4   A         XEND2     B5F8   A         XENTER    ADD8   A         XEQU      B82B   A
XEXP      BA91   A         XFIND     AC7B   A         XFMFLG    00F1   A         XFORM     DEA5   A         XG        AFA2   A
XIF       B604   A         XIJMP     B0AB   A         XITVBV    E462   A         XJMP      B0A5   A         XJSR      B090   A
XL        B245   A         XLIST     AB51   A         XLOAD     ADE8   A         XLOMEM    ADC0   A         XM        B1DC   A
XNEW      A288   A         XNUM      AA88   A         XOPT      B872   A         XORG      B786   A         XPAGE     B6DA   A
XPRINT    AB39   A         XREN      AAFB   A         XREP      AC6A   A         XRTI      B0B6   A         XRTS      B0C2   A
XS        AFF3   A         XSAVE     AE3C   A         XSIZE     ADA0   A         XT        AFFC   A         XTAB      B76B   A
XTITLE    B691   A         XV        B197   A         XWORD     B71B   A         XX        AF9B   A         ZF1       DA46   A*
ZFP       0002   A         ZFRO      DA44   A         ZICB      0020   A         ZPG       0004   A         ZPG1      0080   A
```

```
ZPG2    00E6  A     ZPG3    00FB  A     ZPGX    0040  A     ZPGY    0002  A     ZT1     00AF  A
ZT2     0496  A     ZTEMP1  00F5  A     ZTEMP3  00F9  A     ZTEMP4  00F7  A     ZXLY    DA48  A
!ABS    A910  A     !ADR    A885  A     !ADR2   A890  A     !ANTV   0000  A*    !ASERR  A125  A
!ASYN1  A13B  A     !ASYN2  A161  A*    !ASYN3  A168  A     !BYTE1  A8BE  A     !CLFLG  A00F  A
!COMME  A12A  A     !CS1    A2A2  A     !DATA   A8A8  A     !DEC    A437  A     !DECNU  A810  A
!DEL    A22C  A     !DEL1   A22C  A*    !DLBL1  A0EB  A     !DLBL2  A10C  A     !DLBL3  A102  A
!DSERR  A09D  A     !EDUM   A0B8  A     !ERN1   A2ED  A     !ERNTV  A2E4  A     !ESERR  A097  A
!ESRT   0001  A     !ETXT1  A234  A     !ETXT2  A1DF  A     !ETXT3  A229  A     !ETXT4  A1DC  A
!EXF    A843  A     !EXPX   A847  A     !F1     A83B  A     !FAIL   A355  A     !FEOL   A840  A
!FILES  A878  A     !FS     A4CB  A     !FSTR   A4C9  A     !FSTR1  A511  A     !FSTRM  A4C5  A
!GCHAR  A410  A     !GETAD  A2F8  A     !GETLN  A245  A     !GLNUM  A24F  A     !HEND   A4B3  A
!HEX    A45E  A     !HEX1   A46C  A     !HEX2   A489  A     !HEXD   A45C  A     !HEXIN  A49C  A
!HEXNU  A885  A     !HFAIL  A49B  A     !ICPCR  A38D  A     !INC1   A365  A     !INCCP  A387  A
!IND    A904  A     !KRTN   A5F1  A     !KRTN2  A5EF  A     !LBL1   A54A  A     !LBL2   A559  A
!LBLR1  A3EE  A     !LBLR2  A3F4  A*    !LBLR3  A3FE  A     !LBLRE  A3DC  A     !LERR   A25F  A
!LF     A5A8  A     !LIST   A8E2  A     !LNF    A565  A     !LNF1   A577  A     !LNF2   A584  A
!LNO    A810  A     !LNRAN  A837  A     !LOC    A421  A     !LOC1   A432  A     !LRFAI  A436  A
!NEXT   A2C5  A     !NHEX   A4B2  A     !NOLBL  A132  A     !NXSC   A38E  A     !OA     A828  A
!OADR   A8A0  A     !OADR2  A89F  A     !OBYTE  A8C4  A     !OFILE  A874  A     !OHEX   A8AE  A
!OLNO   A80C  A     !OM     A881  A     !ONO    A8DF  A     !OOPT   A8EA  A     !OP     A863  A
!OPT1   A8DB  A     !OPVAL  A84A  A     !OQA    A825  A     !OR     0002  A     !OWORD  A8B8  A
!PHERR  A0FD  A     !POP    A33B  A     !POP1   A340  A     !PUSH   A314  A     !QSTR   A8C8  A
!RTN    0003  A     !SA     A897  A     !SASM   A82C  A     !SATBU  A819  A     !SBYE   A819  A
!SBYTE  A8BC  A     !SC     A8A3  A     !SCIX   A378  A     !SCLR   A819  A     !SCODE  0008  A
!SCOVF  A3B0  A     !SCR    A8A4  A     !SD     A899  A     !SDBYT  A8B5  A     !SDEL   A808  A
!SDOS   A819  A     !SDR    A897  A     !SEND   A8D2  A     !SENTE  A87D  A     !SEQU   A8D1  A
!SET    A3D2  A     !SETCO  A3A7  A     !SFIND  A814  A     !SG     A896  A     !SHIFT  A47E  A
!SIF    A8D3  A     !SJMP   A8ED  A     !SJSR   A842  A*    !SL     A899  A     !SLIST  A86D  A
!SLNUM  A258  A*    !SLOAD  A878  A     !SLOME  A893  A     !SM     A88F  A     !SMULT  A8F3  A
!SNEW   A819  A*    !SNUM   A871  A     !SOPT   A8D9  A     !SORG   A8D1  A     !SPAGE  A8B2  A
!SPRIN  A86D  A     !SRC1   A5F8  A     !SRC2   A609  A     !SRC3   A615  A     !SRCFN  A621  A
!SRCNF  A623  A     !SRCON  A3C3  A     !SREL   A842  A     !SREN   A871  A     !SREP   A81B  A
!SRFAI  A3D0  A     !SS     A896  A     !SSAVE  A889  A     !SSIZE  A819  A     !ST     A896  A
!ST2    A4E7  A     !ST3    A4EB  A     !ST4    A4FC  A     !ST5    A50F  A     !ST6    A4E9  A
!STAB   A8CD  A     !STFAI  A522  A     !STITL  A8B2  A     !STLEN  A51C  A     !STNUM  A452  A
!STR1   A4B4  A     !STR2   A4D7  A     !STR3   A4DF  A     !SV     A88F  A     !SWORD  A8B5  A
!SX     A897  A     !SYN4   A1EC  A*    !SYN5   A1F2  A*    !SYN5A  A1F9  A*    !SYN6   A1FC  A
!SYN7   A21A  A     !SYNA   A276  A*    !SYNB   A267  A     !SYNCO  A210  A     !SYNEN  A2AD  A
!SYNIN  A218  A     !TAFAI  A3C1  A     !TBIG   A30B  A     !TERMT  A396  A     !TNC1   A44D  A
!TNC3   A448  A     !TSTAL  A3B6  A     !TSTOR  A36A  A     !TSTSU  A2D8  A     !TV1    A536  A
!TVFAI  A548  A     !TVS1   A596  A     !VALUE  A84E  A     !VEXP   0007  A     !APR1   B334  A
!APR2   B33E  A     !BBACK  B3BD  A     !C1     AF70  A*    !C2     AF68  A     !C3     AF66  A
!C4     AF7E  A     !CADR   B286  A     !CDONE  AF8D  A     !CERR   AF8A  A*    !CONT1  AA2B  A
!CONT2  AA61  A     !CONT3  AA5D  A     !CONT4  AA68  A*    !D1     AEC9  A     !D2     AED3  A
!D3     AEC1  A     !D4     AEF5  A     !D5     AEE0  A     !DDONE  AF02  A     !DDUM   B3CF  A
!DIS2   B2AF  A     !DIS3   B2BA  A     !DIS4   B2CB  A     !DISEN  B35F  A     !DL     B274  A
!DLDON  B296  A     !DM1    B43C  A     !DM2    B410  A     !DRTN   AB38  A     !DTST   AEED  A*
!EXP1   A99D  A     !EXP2   A9A8  A     !EXP3   A9CE  A     !EXP4   A9EF  A     !EXP5   A9F4  A
!EXP6   A9F8  A     !EXP7   A9FF  A     !FA2    B228  A     !FA3    B239  A     !FAFAI  B22F  A
!FDONE  ACAE  A     !FIND1  ACA5  A     !FIND2  AC95  A     !FMN    B30C  A     !FND    AD6B  A
!FR     AC82  A     !GO     AFA5  A*    !GOBJ   AE3B  A*    !GPC    B134  A     !INB    ABEF  A
!LEND   AE96  A     !LEOF   AE1F  A     !LEOF1  AE21  A     !LERR   AE27  A     !LGO    AB8B  A*
!LIST1  AB7E  A     !LIST2  AB74  A     !LM1    ABDB  A     !LMERR  ADD3  A     !LOAD   AE0D  A
!LOAD2  ADF9  A     !LPASS  AE97  A     !LRAN1  AC53  A     !LRAN2  AC3B  A*    !LRAN3  AC51  A
!LRAN4  AC4F  A     !LRTN   AB9C  A     !LS1    AD45  A     !LS2    AD4D  A     !MASE1  B500  A*
```

```
                    !GO     AFA5  A*    !GOBJ   AE3B  A*    !GPC    B134  A     !INB    ABEF  A
!LEND   AE96  A     !LEOF   AE1F  A     !LEOF1  AE21  A     !LERR   AE27  A     !LGO    AB8B  A*
!LIST1  AB7E  A     !LIST2  AB74  A     !LM1    ABDB  A     !LMERR  ADD3  A     !LOAD   AE0D  A
!LOADP  ADF9  A     !LPASS  AE97  A     !LRAN1  AC53  A     !LRAN2  AC3B  A*    !LRAN3  AC51  A
!LRAN4  AC4F  A     !LRTN   AB9C  A     !LS1    AD45  A     !LS2    AD4D  A     !MASE1  B500  A*



PAGE   309     SHEP   ATARI EDITOR
END OF BASIC

!MASER  B4FB  A     !MEOF   B217  A     !MINI   B4EB  A     !MINI2  B4F5  A     !MOVE   B1DF  A
!MOVL   ABC6  A     !MPR    B310  A     !NCLR   ADE5  A     !NDONE  AAC0  A     !NF     B43E  A
!NF1    B43F  A     !NFND   AD6C  A     !NOR    B04F  A     !NOR2   B056  A     !NUM1   AAB3  A
!NUM2   AA9A  A     !OPF    B478  A     !OPS1   B46D  A     !PRAD1  B391  A     !PRR    B376  A*
!PSET   AAF9  A     !Q1     ACB6  A     !Q2     ACC6  A     !RCON   AD04  A     !REN2   AB07  A
!REN3   AB38  A     !REP    AD07  A     !REP2   AD28  A     !REP3   AD37  A     !SAVE   AE5C  A
!SCAN1  B2F7  A     !SCAN2  B301  A     !SEOF   AE6B  A     !SET1   AD84  A     !SETA   AD80  A*
!SETIT  AD86  A     !SIJMP  B448  A     !STRM1  AC03  A     !TFAIL  AE84  A     !TI     AFFF  A
!TLN    AE77  A     !TPASS  AE84  A     !V1     B19A  A     !V2     B1D0  A     !V3     B1D3  A*
!VEOF   B1DB  A     !XPDON  AB50  A     !ABRK   B58E  A     !ADONE  B58E  A     !ADR1   B9C7  A*
!ADRER  B9B6  A     !ADRR   B9C5  A     !ADUM   B5DB  A     !AL1    BBDD  A*    !AL2    BBE9  A
!AL3    BBF6  A     !AL4    BBEC  A     !AL5    BC00  A     !AL6    BBD7  A*    !ASDON  B5D9  A
!ASYNE  B5D6  A     !AUTO   BCDE  A     !AXM1   B968  A     !AXM2   B96B  A     !AXM4   B969  A
!AXM5   B955  A     !AXM6   B965  A     !BACK   B8EB  A     !BERR1  B765  A     !BYT1   B750  A
!BYT2   B745  A     !BYT3   B75B  A     !DOLP   BAF8  A     !DVJN   BB12  A     !EDONE  B86B  A
!EDUM   BAB7  A     !EERR   B86C  A     !END    B602  A     !ERRB   BB5F  A*    !EXDON  BAB2  A
!GEXP   B93C  A     !GEXP1  B93A  A     !GLTXT  B669  A     !IF1    B612  A     !IF2    B620  A
!IF3    B62B  A     !IF4    B655  A     !IFF    B639  A     !IFP    B642  A     !IJMP   B909  A
!JSR    B907  A     !MDRT   BB3C  A     !MDSU   BB34  A     !MILP   BB29  A     !MOLP   BB19  A
!NL     BCA6  A     !OC1    BB80  A     !OC2    BB8E  A     !OC3    BBA7  A     !OC4    BBBD  A
!OC5    BBCB  A     !OC6    BBB4  A     !OC7    BBB7  A     !OD     B7F3  A     !OFF1   B8A5  A
!OKB    BB62  A     !ON     B88A  A     !OP2    B7C6  A     !OPT    B8AF  A     !ORG    B79A  A
!ORGA   B7A7  A     !ORGB   B7D2  A     !PAD    BC8C  A     !PARM2  B54C  A     !PARM3  B55D  A
!PDONE  B56A  A     !PIN1   BC55  A     !PO     B80A  A     !PO1    BC10  A     !PO2    BC1D  A
!PODON  BC27  A     !PRB    BC85  A     !PRPL1  BC6B  A     !PRPL2  BC75  A     !PT     B714  A
!PT2    B6C7  A     !PTDON  B6D7  A     !PTEXP  B6BB  A     !PTMOV  B6C1  A     !PTPR   B710  A
!PTPR1  B705  A     !RES    BB51  A     !SP1    B999  A     !SPEND  B9A1  A     !TAB    B76F  A
!TL1    BCA5  A     !TSTE1  BCC5  A     !UPERR  BD23  A     !VALID  B982  A     !VALOK  B8F5  A
!VERR   B8F2  A     !WORD   B71D  A     !WORD2  B71F  A     !WORD3  B729  A     !XEXP1  BA9C  A
!ASMER  BEF5  A     !EDERR  BEF2  A     !ERTN   BEE0  A     !ERTN1  BEE7  A     !GS2    BE92  A
!GS3    BEAC  A     !GSRT1  BEAB  A     !MSG1   BFA2  A     !PDUM   BD51  A     !PEDON  BF25  A
!PENUM  BF0D  A     !PRI1   BF3E  A     !PRIH   BF65  A     !PRNON  BEF5  A     !PRT1   BF01  A
!SIO1   BDD3  A     !SIO2   BDE1  A     !SIO2A  BDF9  A     !SIO2B  BDF6  A     !SIO5   BDFA  A
!SIO6   BDF1  A     !SIO7   BDDC  A     !TB2    BE88  A     !ADD1   DA98  A     !ADD2   DAA7  A
!ADDEN  DAA4  A     !CVB1   DC79  A*    !CVBYT  DC76  A     !CVFRO  DC70  A     !DECIN  DCC1  A
!DP     D85A  A     !EF1    D94F  A     !EF2    D96B  A     !EF3    D972  A     !EF4    D97A  A
!EFORM  D920  A     !EMIN   D8AB  A*    !EPL    D969  A     !EPLUS  D8AD  A     !EROV   DB26  A
!ERVAL  DA42  A     !EVEN   D8CE  A     !EXIT   D8B6  A     !EXIT1  D8C3  A     !EXP    D863  A
!EXPO   D91A  A     !EXP1   D89B  A     !EXP2   D86C  A     !EXP3   D88E  A     !F1     DD0B  A
!F2     DD11  A     !FADON  D9A9  A     !FAIL   DB65  A     !FAIL2  DB87  A     !FN1    DCB4  A
!FN2    DCB8  A     !FN3    DCA6  A     !FN4    D99C  A     !FN5    D988  A     !FN6    D914  A
!FNZER  DCA4  A     !FPI1   D9EA  A     !FRA    DD13  A     !FRA1   DD19  A     !FRADD  DA66  A
!FRD1   DB4E  A     !FRD2   DB70  A     !FRM    DAF1  A     !FRM1   DAF7  A     !FRM2   DB01  A
!FRM3   DB09  A     !FRS1   DB74  A     !FRS2   DB52  A     !GCHR1  DB9D  A     !GETCH  DB94  A
!GETDI  DCB9  A     !IFP1   D9B8  A     !IFP2   D9BE  A     !ILSHF  DA5A  A     !IN1    D818  A
!IN2    D81C  A     !INCE   D837  A     !INCE2  D83E  A     !IND2   D8E5  A     !INDON  D8E4  A
!MINUS  D856  A     !MV1    DD36  A     !MV2    DD2A  A     !NALIG  DA8E  A     !NDONE  DC38  A
!NIB1   DBED  A     !NIBS   DBEF  A     !NOINC  D94C  A     !NON1   D842  A     !NON2   D8A3  A
!NONUM  D841  A     !NORM   DC0A  A     !NOTE   D8B2  A     !NR     DA38  A     !NSH    DC10  A
!NSWAP  DA85  A     !NXTB   DB13  A     !NXTB1  DC64  A     !NXTQ   DB43  A     !NXTQ1  DB45  A
!ROUND  DA24  A     !RSH    DC40  A     !RSH1   DC48  A     !RSH2   DC46  A     !RTFAI  DBCF  A
!RTPAS  DBE2  A     !SB1    DBA5  A     !SBRTS  DBAC  A     !STCHA  DC9F  A     !STNUM  DC9D  A
!SUB    DAB3  A     !SUB1   DAB6  A     !SUB2   DAC5  A     !SUB3   DACE  A     !SWAP   DA77  A
!TSNFA  DBD0  A     !TST1   DC9C  A     !TSTBI  DC28  A     !TSTCH  DBBB  A     !TSTDP  DC93  A
!TSTN   DBDB  A     !TSTN1  DBD2  A     !TSTUN  DC31  A     !ZF2    DA4A  A
  47 ERROR LINES  203  292  294  295  298  299  304
SOURCE CK = 5C08 OBJ CK = 0000
 4 LOCAL REGIONS (63 MAX)
```

```
REMAINING TABLE SPACE = 0E0B
IN RANGE FLAG COUNT=    0
DISC SOURCE FILE (HEX)= 0300-048A
```

```
 A
 *
 I
                                     END OF EXCLUDES
                                     END OF INCLUDES
 !ABCDEFGHIJKLM*
 , *-*/.()=[illegible]
 Y YYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
 72
 0300                                LAST COLUMN CARD


 !ABRK      5727    5746*   6174
 !ABS       3001*
 !ADD1      8794*   8799
 !ADD2      8802    8808*
 !ADDEN     8776    8803*
 !ADONE     5735    5745*
 !ADR       2707*
 !ADR1      6631*
 !ADR2      2727*
 !ADRER     6570    6621*
 !ADRR      6547    6624    6629*
 !ADUM      5776    5804*
 !AL1       7198*
 !AL2       7201    7206*   7212
 !AL3       7210    7213*
 !AL4       7204    7208*
 !AL5       7217    7219*
 !AL6       7192*
 !ANTV      2468*
 !APR1      5216*   5226    5234    5236
 !APR2      5222*   5286
 !ASDON     5793    5800*
 !ASERR      675     735*    763     765     812
 !ASMER     7868    7879*
 !ASYN1      729     753*
 !ASYN2      776*
 !ASYN3      772     781*
 !ASYNE     5790    5798*
 !AUTO      7411    7416*
 !AXM1      6541    6545    6554*
 !AXM2      6559*   6572
 !AXM4      6514    6532    6535    6556*
 !AXM5      6523    6528    6537*
 !AXM6      6543    6546*
 !BACK      6449    6454*
 !BBACK     5323    5330*
 !BERR1     6104    6126*
 !BYT1      6099    6112*
 !BYT2      6105*   6109
 !BYT3      6110    6118*
 !BYTE1     2852*
 !C1        4395*
 !C2        4383    4391*   4402    4410
 !C3        4375    4389*
```

```
!C#        4399     4404*
!CADR      5075     5081*
!CDONE     4393     4397     4420*
!CERR      4415*
!CLFLG      499*     502
!COMME      670      738*     758
!CONT1     3245*    3255
!CONT2     3275     3277     3283*    3286
!CONT3     3280*    3290
!CONT4     3288*
!CB1       1064*    1069
!CVB1      9328*
!CVBYT     9326*    9350
!CVFRO     8373     8401     9318*
!D1        4270*    4311
!D2        4277*    4306     4308     4310
!D3        4236     4250     4258*
!D4        4294     4299     4302*
!D5        4283     4285*
!DATA      2797*
!DDONE     4290     4295     4300     4315*
!DDUM      5121     5343*
!DEC       1434*    2503
!DECIN     8384     8504     9424*
!DECNU     2501*
!DEL        880      951*
!DEL1       952*
!DIS2      5109     5112*
!DIS3      5111     5119*
!DIS4      5133*    5139
!DISEN     5210     5218     5241*
!DL        5047     5056     5068*    5078     5084     5088     5089
!DLBL1      680      689*
!DLBL2      712      715*
!DLBL3      688      694      702      708*
!DLDON     5070     5085     5090*
!DM1       5447     5450     5452*
!DM2       5419*    5423
!DOLP      6968*    6988
!DP        8186     8205*
!DRTN      3466     3472*
!DSERR      575      578      614*
!DTST      4296*
!DVJN      6980     6986*
!EDERR     7874     7876*
!EDONE     6337     6350*
!EDUM       643      647*    6880     6899*
!EERR      6311     6355*
!EF1       8416     8437*
!EF2       8456     8460*
!EF3       8466*    8471
!EF4       8469     8473*
!EFORM     8364     8366     8399*
!EMIN      8272*
!END       5832     5838*
!EPL       8446     8458*
!EPLUS     8267     8274*
```

```
!ERN1      1113    1115*
!ERNTV     1093    1110*
!EROV      8880    8957*   8979    8989
!ERRB      7076*
!ERTN      7856    7859    7863*
!ERTN1     7867    7869*
!ERVAL     8587    8589    8609    8617    8619    8628    8639    8671*
!ESERR      542     544     611*
!ESRT      2470*   2503    2512    2529    2530    2624    2626    2628    2630    2683    2708    2717    2872    2899
!ETXT1      954     956*
!ETXT2      900*    906
!ETXT3      943     945*
!ETXT4      884     889     895*
!EVEN      8317    8320*
!EXDON     6873    6891*
!EXIT      8191    8207    8260    8291*
!EXIT1     8304    8305    8311*
!EXP       1297    1299    2601*   8188    8212*
!EXPO      8361    8391*
!EXP1      3127    3131*   8249    8255*
!EXP2      3126    3129    3130    3137*   8220*   8276
!EXP3      3160*   3170    8228    8247*
!EXP4      3184*   3198
!EXP5      3188*   3191
!EXP6      3181    3190*
!EXP7      3182    3196*
!EXPX      2605*
!F1        2581*   9495    9503*
!F2        9499    9508*
!FA2       4988    4992    4996*
!FA3       5013    5015*
!FADON     8502    8509*
!FAFAI     4989    4993    5001*
!FAIL      1101    1108    1191    1198*   1201    1212    1216    9028    9036*
!FAIL2     9061    9069*
!FDONE     3750    3776    3779    3792    3798*
!FEOL      2590*
!FILES     2681*
!FIND1     3785    3794*
!FIND2     3783*   3829    3922
!FMN       5172    5188*
!FN1       9390    9400*
!FN2       9392    9403*
!FN3       9387*   9394
!FN4       8385    8489    8500*
!FN5       8382    8486*
!FN6       8381    8383*
!FND       3947    3964*
!FNZER     8375    8403    9384*
!FPI1      8604*   8643
!FR        3756    3768*
!FRA       9505    9512*
!FRA1      9517*   9524
!FRADD     8749*   8771
!FRD1      8993    9012*   9032
!FRD2      9045*   9065
!FRM       8889*   8935
```

```
!FRA1       8899*    8900
!FXM2       8901     8907*
!FXM3       8912*    8921
!FMU1       9053*    9060
!FRMC       9020*    9025
!FS         1571     1577*
!FSTR       1575*    2680     2897
!FSTR1      1581     1632*
!FSTNM      1569*    2717
!GCHAR      1401*    2626
!GCHR1      6244     9071     9075*
!GETAD      1096     1099     1122*
!GETCH      8141     8215     8225     8275     9088*    9136     9154     9159
!GETDI      8633     8648     9414*
!GETLN       530      973*
!GEXP       6510     6517*
!GEXP1      6511     6515*
!GLNUM       975      981*
!GLTXT      5913     5921*
!GO         4451*
!GOBJ       4130*
!GPC        4777     4782*
!GS2        7801*    7821
!GS3        7807     7809     7812     7818*
!GSRT1      7803     7808     7813     7815*
!HEND       1533     1538     1542*
!HEX        1465*    2708
!HEX1       1481*    1506
!HEX2       1492     1508*
!HEXD       1462*    2624
!HEXIN      1476     1491     1527*
!HEXNU      2706*
!HFAIL      1477     1500     1517*
!ICPCR      1234     1236*
!IF1        5847     5849     5853     5856*
!IF2        5866*    5884
!IF3        5872*    5879
!IF4        5868     5902*
!IFF        5880*    5890     5892
!IFP        5874     5888*
!IFP1       8536*    8553
!IFP2       8542*    8550
!IJMP       6481     6484*
!ILSHF      8608     8616     8618     8713*
!IN1        8136*    8160     8163     8175     8193     8203     8210
!IN2        8133     8140*
!INB        3630*    3638
!INC1       1203     1210*
!INCCP      1209     1211     1232*    1244     1355
!INCE       8149     8169*
!INCE2      8172     8174*
!IND        2987*
!IND2       8327     8339*
!INDON      8332     8337*
!JSR        6478     6482*
!KRTN       1780     1791     1795     1799     1803     1807     1813     1817     1821*
!KRTN2      1811     1819*
```

```
!LBL1     1676     1683*
!LBL2     1674*    1699
!LBLR1    1370     1376*
!LBLR2    1381*
!LBLR3    1379     1387*
!LBLRE    1364*    2630
!LEND     4211     4212     4217*
!LEOF     4102*    4126
!LEOF1    4104*    4164
!LERR      982      991*    4077     4080     4109*
!LF       1697     1748*
!LGO      3540*
!LIST     2924*
!LIST1    3511     3531     3533*
!LIST2    3517     3528*
!LM1      3610*    3618
!LMERR    4044     4052*
!LNF      1695     1701*
!LNF1     1705     1712*
!LNF2     1722*    1725
!LNO      2502*
!LNRAN    2573*
!LOAD     4091*    4098
!LOAD2    4082*    4093
!LOC      1417*    2628
!LOC1     1345     1359     1395     1410     1425*    1455
!LPASS    4216     4219*
!LRAN1    3705     3710*
!LRAN2    3697*
!LRAN3    3683     3696     3708*
!LRAN4    3700     3706*
!LRFAI    1368     1428*
!LRTN     3539     3542     3546     3550*
!LS1      3937*    3962
!LS2      3943*    3958
!MABE1    5652*
!MABER    5620     5625     5629     5649*
!MDRT     7009     7032*
!MDBU     6966     7001     7027*
!MEOF     4934     4936     4976*
!MILP     7015*    7020
!MINI     5613     5638*
!MINI2    5615     5644*
!MINUS    8195     8201*
!MDLP     7003*    7011     7021
!MOVE     4932*    4944     4946
!MOVL     3589*    3593
!MPR      5192*    5200
!MSG1     8072*    8082
!MV1      9549*    9554
!MV2      9533*    9538
!NALIG    8774     8783*
!NCLR     4063     4066*
!NDONE    3344     3377*    9209     9244     9247*
!NEXT     1090*    1100     1107     1169     1192     1228
!NF       5408     5455*
!NF1      5421     5428     5457*
```

```
!NFND      3921      3956      3966*
!NNEX      1531      1536      1540*
!NIB1      9177      9181*
!NIBB      9183*     9193
!NL        7369      7374*
!NOINC     8426      8431*
!NOLBL      672       744*
!NON1      8142      8184*
!NON2      8216      8265*
!NONUM     8121      8178*
!NOR       4597*     4601
!NOR2      4603*     4737
!NORM      9206*     9226
!NOTE      8269      8282*
!NR        8650      8661*
!NSH       9213*     9218
!NSWAP     8757      8773*
!NUM1      3357      3369*
!NUM2      3355*     3368
!NXSC      1091      1123      1125      1199      1243*
!NXSC1     1247      1270      1281*
!NXTB      8919      8925*
!NXTB1     9301*     9306
!NXTQ      8996*     9073
!NXTQ1     9002*     9008
!OA        2543*
!OADR      2777*
!OADR2     2775*
!OBJOV     7103      7105*
!OBYTE     2862*
!OC1       7127*     7191
!OC2       7126      7136*
!OC3       7154*     7174
!OC4       7168      7170*
!OC5       7139      7144      7181*
!OC6       7157      7163*
!OC7       7161      7166*
!OD        6231      6235      6246*
!OFF1      6388      6392*
!OFILE     2669*
!OHEX      2807*
!OKB       7071      7074      7078*
!OLNO      2493*
!OM        2698*
!ON        6369      6377*
!ONO       2922*
!OOPT      2933*
!OP        2635*
!OP2       6200      6213*
!OPF       5514      5522*
!OPS1      5512*     5518
!OPT       6383      6398*
!OPT1      2913*
!OPVAL     2612*
!OGA       2539*
!OR        2472*     2496      2542      2546      2560      2563      2567      2576      2584      2615      2623      2625      2627      2629      2637
           2639      2641      2643      2659      2672      2701      2765      2779      2800      2810      2819      2838      2855      2865      2923
```

```
               2925    2926    2927    2936    2948    2964    2971    2977    2992    3005    3009
 !ORG          6169    6176*
 !ORGA         6179    6183*
 !ORGB         6221    6227*
 !OWORD        2835*
 !PAD          7326    7327    7348*
 !PARM2        5680    5702*
 !PARM3        5705    5711*
 !PCHR         1252    1267    1272*
 !PDONE        5678    5700    5710    5724*
 !PDUM         7509    7517*
 !PEDON        7904    7914*
 !PENUM        7902*   7912
 !PHERR         698     704*
 !PHEX         1254    1256    1258    1260    1262    1265    1274*
 !PIN1         7285*   7291
 !PO           6259    6269*
 !PO1          7235*   7247
 !PO2          7240    7242*
 !PODON        7231    7245    7248*
 !POP          1104    1174*   1215
 !POP1         1176    1180*
 !PRAD1        5278    5283*
 !PRB          7344*   7347
 !PRI1         7934*   7943
 !PRIH         7986    7990*
 !PRNON        7878*   7889
 !PRPL1        7308*   7312
 !PRPL2        7305    7306    7313*
 !PRR          5261*
 !PRSP         1273    1275*
 !PRT1         7891    7893*
 !PSET         3394    3405    3418*
 !PT           6054    6056*
 !PT2          5997*   6003
 !PTDON        5964    6007*
 !PTEXP        5977    5986*
 !PTMOV        5976    5985    5992*
 !PTPR         6046    6052*
 !PTPR1        6020    6035*
 !PUSH         1121    1127    1147*   1301
 !Q1           3812*   3827
 !Q2           3817    3819*
 !QSTR         2870*
 !RCON         3869    3876*
 !REN2         3437*   3452
 !REN3         3439    3471*
 !REP          3868    3874    3877*
 !REP2         3909*   3917
 !REP3         3907    3919*
 !RES          7057    7059*
 !ROUND        8592    8647*
 !RSH          9261    9266*
 !RSH1         9273*   9278
 !RSH2         9271*   9284
 !RTFAI        9147*   9157    9161
 !RTN          2473*   2489    2497    2504    2522    2534    2547    2569    2577    2586    2593    2604    2608    2616    2631
```

```
           2645      2665      2673      2684      2694      2702      2709      2719      2737      2755      2769      2780      2793      2803      2811
           2820      2831      2839      2848      2858      2866      2873      2889      2900      2909      2917      2923      2929      2937      2950
           2983      2997      3011
 !RTPAS    9137      9155      9160      9164*
 !SA       2357      2753*
 !SASM     2349      2558*
 !SATBU    2350      2520*
 !SAVE     4152*     4159
 !SBI      9108*     9111
 !SBRTS    9109      9113*
 !SBYE     2347      2518*
 !SBYTE    2379      2846*
 !SC       2368      2787*
 !SCAN1    5167*     5181
 !SCAN2    5174*     5177
 !SCIX     1220      1222*
 !SCLR     2335      2516*
 !SCODE    1291      2475*     2603      2970      2981      2988      2993      3002      3006      3010
 !SCOUT    1272      1276*
 !SCOVF    1311      1313*
 !SCR      2359      2789*
 !SD       2364      2762*
 !SDBUG    1246      1279*
 !SDBYT    2380      2827*
 !SDEL     2336      2485*
 !SDOS     2348      2519*
 !SDR      2360      2752*
 !SEND     2381      2888*
 !SENTE    2338      2690*
 !SEOF     4154      4162*
 !SEQU     2385      2885*
 !SET      1292      1354*
 !SET1     3979      3988*
 !SETA     3984*
 !SETCO     724       726       728       746       768       770       779       824       896       903      1307*     1389      1392      1404      1407
           1421      1423      1426      1448      1453      1510      1512      1514      1611      1614      1618      1627
 !SETIT    3983      3986      3990*
 !SFIND    2337      2511*
 !SG       2361      2747*
 !SHIFT    1498*     1503
 !SIF      2375      2896*
 !SIJMP    5399      5465*
 !SIO1     7645      7649*
 !SIO2     7653      7659*
 !SIO2A    7665      7670      7673*
 !SIO2B    7663      7671*
 !SIO5     7661      7677*
 !SIO6     7654      7668*     7679      7682
 !SIO7     7655*     7695
 !SJMP      868       870      2944*
 !SJSR     2600*
 !SL       2365      2763*
 !SLIST    2334      2656*
 !SLNUM     986*
 !SLOAD    2339      2680*
 !SLOME    2346      2734*
 !SM       2358      2724*
```

```
!SMULT      802      804     2957*
!SNEW      2515*
!SNUM      2341     2662*
!SOPT      2382     2907*
!SORQ      2384     2886*
!SP1       6592     6599*
!SPAGE     2377     2817*
!SPEND     6587     6597     6601     6606*
!SPRIN     2342     2655*
!SRC1      1850*    1878
!SRC2      1860*    1871
!SRC3      1864     1869*
!SRCFN     1874     1880*
!SRCNF     1856     1883*
!SRCON     1293     1336*
!SREL       851      853     2599*
!SREN      2343     2661*
!SREP      2344     2528*
!SRFAI     1340     1344     1347*
!SS        2363     2748*
!SSAVE     2340     2715*
!SSIZE     2345     2517*
!ST        2362     2749*
!ST2       1580     1603*
!ST3       1591     1608*
!ST4       1619*    1628
!ST5       1626     1630*
!ST6       1567     1605*
!STAB      2383     2880*
!STCHA     8441     8461     9358     9373*
!STFAI     1565     1624     1638     1647*
!STITL     2376     2816*
!STLEN     1636     1639*
!STNUM      989     1451*    8479     8482     9340     9346     9371*
!STR1      1558*    2512     2529
!STR2      1588*    2530
!STR3      1595*    2872
!STRM1     3650*    3655
!SUB       8787     8818*
!SUB1      8822*    8827
!SUB2      8829     8835*
!SUB3      8844*    8849
!SV        2366     2725*
!SWAP      8763*    8770
!SWORD     2378     2828*
!SX        2367     2751*
!SYN4       908*
!SYN5       913*
!SYN5A      917*
!SYN6       915      919*
!SYN7       937*     941
!SYNA      1012*
!SYNB      1003*    1006
!SYNCO      922      931*
!SYNEN      608      811     1075*
!SYNIN      921      929      936*
!TAB       6137*    6147
```

```
!TAFAI    1326      1330*
!TB1      7773      7776*
!TB2      7781      7784*
!TBIC     1137*     1153
!TERMT    1106      1290*
!TFAIL    4192      4197*
!TI       4532*     4536
!TLI      7360      7364      7367      7371*
!TLN      4178      4180*
!TNC1     1441      1447*
!TNC3     1439      1442*
!TPASS    4191      4196*
!TSNFA    9123      9149*
!TSTI     9356      9359*
!TSTAL    1321*     1665      1670
!TSTPI    9208      9231      9236*
!TSTCH    8120      9133*
!TSTOP    9327      9354*
!TSTE1    7385      7389      7396*
!TSTN     9140      9158*
!TSTN1    9142      9144      9153*
!TSTOR    1207      1214*
!TSTSU    1097      1103*
!TSTUN    9240      9242*
!TV1      1668*     1671      1673      1675
!TVFAI    1660      1666      1678*     1709
!TVS1     1736*     1740
!UPERR    7466      7469*
!V1       4873*     4917      4919
!V2       4888      4912*
!V3       4915*
!VALID    6562      6565      6574*
!VALOK    6452      6459      6464*
!VALUE    2621*
!VEOF     4878      4885      4921*
!VERR     6451      6457      6462*
!VEXP     2474*     2829      2856      2881      2883      2887      2897      2946      2949      2962      2969      2975
!WORD     6068      6072*
!WORD2    6075*     6088
!WORD3    6079      6083*
!XEXP1    6870*     6884
!XPDON    3485      3490*
!ZF2      8690*     8694
ABORT     7114*     7148
ABRK      3135      5744*     7119
ABS       6638*     6653      6654      6693      6696      6699      6702      6705      6708      6711      6714      6717      6732
ABSX      6644*     6653      6654      6693      6696      6702      6734
ABSY      6645*     6656      6659      6662      6665      6668      6671      6674      6677      6698      6733
ACU       6640*     6654      6728
ADD       6914      6936*
ADDRON     399*     5063
ADISPL     269*      784      5762      7271
ADRBIT    6560      6563      6724*
ADSTBL     838       840      2374*
ADXTBL    3070*     5821      5823
AFP        974      1438      8116*
AINIT     5597      5662      6773*
```

```
AINIT2    5742   6789*
ALIST     7185*
ALIST1    6346   7197*
AMODE      349*  5669
AND       6918   6923*
ARDBV       14*    17     491   1021   4165   7102   7558   7770
APHM        67*  3173    3175   3257   3259
AQ1FLG     110*  3791    3805   3920   3991
ARES1     5749   6840*   7657
ARES2     6778   6850*
ARESET    5645   6843*
ASCIN     8118*
ASDIR      834*  2392
ASIMP      819*  2393
ASM1A     5639   5759*   5900
ASM1B     5919   5934*
ASM1C     5938   5943*
ASM1D     5923   5940*
ASM1L     5729   5755*
ASM3      5725*  5732    5743
ASMISC     863*  2395
ASMR      5748*
ASMULT     801*  2396
ASREL      850*   866    2394
ASTBL      791    793    2390*
ASYNTA     663*  5765
ATARIV      12*    39     484    516   2301   3172   3256   3300   3314   3323   3502   3551   4455   4476   4829
          7637   7850    8094   9859
ATCOEF    9845*  9856
ATR1        15*
AUTO      7402*  7483
AUTOFF     353*   891    7412
AUTOFL     134*   883     892    942   3341   6847   7413   7482
AUTON      352*  3340
AXCOM     3058   5791*   5851
AXDIR     3059   5816*
AXIMP     3060   6422*
AXMISC    3062   6473*
AXMULT    3063   6496*
AXREL     3061   6431*
AXTBL     3057*  5807    5809
BELL        36*    55*    624   7894
BININT     191*   983     984
BIT8      6115   6544    7069*
BLKCNT     162*  7343    7346
BRKBYT      27*    60*   4482   4490   7647   7776   7780   7783
BRKINT    4456*
BRKPAT    4831*
BRKVEC      59*  3325    3327
BUG1      3331*  5646
BUGMSG     562    563     587*
BYELOC      63*  3301
BYTE*     1919
C         1901*  1904*   1904   1905   1906   1909*  1909   1912*  1912   1915*  1915   1918*  1918   1921*  1921
          1924*  1924    1927*  1927   1930*  1930   1933*  1933   1936*  1936   1938   1940*  1940   1941   1942
          1948*  1948    1951*  1951   1954*  1954   1957*  1957   1960*  1960   1963*  1963   1966*  1966   1969*
          1969   1972*   1972   1975*  1975   1978*  1978   1981*  1981   1984*  1984   1987*  1987   1990*  1990
```

```
              1993*    1993     1996*    1996     1999*    1999     2002*    2002     2005*    2005     2008*    2008     2011*    2011     2014*
              2014     2017*    2017     2019     2021*    2021     2022     2026*    2026     2029*    2029     2032*    2032     2035*    2035
              2038*    2038     2041*    2041     2044*    2044     2046     2048*    2048     2049     2051     2054*    2054     2055     2058
              2060*    2060     2061     2065*    2065     2068*    2068     2071*    2071     2074*    2074     2077*    2077     2080*    2080
              2083*    2083     2086*    2086     2089*    2089     2092*    2092     2095*    2095     2098*    2098     2101*    2101     2104*
              2104     2107*    2107     2110*    2110     2113*    2113     2116*    2116     2119*    2119     2122*    2122     2125     2126
              2127     2128*    2128     2129     2131     2134*    2134     2135     2138*    2138     2139     2142*    2142     2143     2146*
              2146     2148     2149     2153     2154*    2154     2157*    2157     2160*    2160     2163*    2163     2165*    2165     2168*
              2168     2172*    2172     2175*    2175     2178*    2178     2181*    2181     2184*    2184     2187*    2187     2190*    2190
              2193*    2193     2196*    2196     2199*    2199     2201     2203*    2203     2204     2206     2209*    2209     2210     2213*
              2213     2216*    2216     2219*    2219     2222*    2222     2225*    2225     2228*    2228     2231*    2231     2234*    2234
              2237*    2237     2238     2241*    2241     2243     2245*    2245     2246     2247     2250*    2250     2251     2254*    2254
              2255     2256     2259*    2259     2260     2263*    2263     2264     2267*    2267     2268     2271*    2271     2272     2275*
              2275     2276     2279*    2279     2280     2283*    2283     2284     2287*    2287     2288     2291*    2291     2292     2293
              2296*    2296     2297     2305*    2305     2306     2309*    2309     2310     2313*    2313     2314     2317*    2317     2318
              2321*    2321     2322     2324     2326*    2326     2327
  CA          2206*    2545     2982     3782
  CAND        2264*    2642
  CCOM        2306*    2494     2540     2544     2583     2585     2671     2699     2730     2776     2798     2808     2836     2863     2882
              2884     2934     2989     2995     3003     3007     4398     5679     5704     6087     6120
  CDELIM      2310*    2728     2790
  CDISPL       270*     822     7278
  CDIV        2260*    2638
  CDO          394*    3823
  CDOL        2284*    2624
  CDQ         2268*    2871
  CDUM        2327*    4558     5458
  CEASM       2127*
  CEBUG       1816     2243*
  CEDIR       1790     1938*
  CEEDIT      1812     2201*
  CEIMP       1794     2019*
  CEJECT      2143*    2927
  CEMISC      1802     2058*
  CEMNEM      2126*
  CEMULT      1806     2125*
  CEOL         823     2297*    2488     2521     2533     2568     2592     2658     2664     2693     2736     2754     2764     2768     2792
              3393     3404     3530     3695     3978     4062     4235     4249     4396     4776     5046     5055     5677     5699     5709
              5963     6400
  CEOP        2324*
  CEOPT       2153*
  CEREL       1798     2046*
  CERR        2139*    2926
  CFILE       2293*    2682     2716     3510
  CIFDEL      2322*    2898
  CIMM        2292*    2968
  CIO           28*      47*     492     7761
  CIX          216*     536      569      667      755      759      783      821      877      901      904      978     1008     1084     1156
              1218     1225     1250     1253     1323     1436     1444     1515     1562     1582     1588     1604     1620     1621     1631
              1662     1669     1685     1703     1706     1713     1727     1732     1742     1750     1778     1853     3608     3611     3612
              3629     3635     3636     3740     3742     5617     5869     5876     5877     5895     5936     5939     6287     7256     7303
              8213     8243     8284     8295     8405     8426     8429     8440     8442     9090     9097     9105     9113     9119     9134
              9166
  CJMP        2055*    5467
  CJSR         865     2051*
  CKREP       3788     3804*
  CLALLI      7736*    7739
```

```
 CLALL7     3330    7734*
 CLASF      1788*   5118
 CLFILE     7648    7662    7669    7700*
 CLIST      2149*   2928
 CLOC       2256*   2628
 CLPRN      2276*   2945    2961    6477
 CLR1       1051    1063*   5741    6854
 CLRMEM      494    1020*
 CLSALL     5747    7705    7732*
 CLSYSD     3522    4106    7713*   7737
 CM         2210*   2700
 CMINUS     2251*   2644
 CMUL       2255*   2636
 CNO        2131*   2923    6365
 CNODO       395*   3826
 CNTON       401*   5042
 COBJ       2135*   2925    6368    6379    6387    6394
 COLDST      475     483*   8098    9863
 COMEXE      547     582     639*
 COMSYN      543     577     594*
 CONTLO      968    3218*   5984
 CONTRA     3220*
 CONTXT      934    3215*   3877
 COPTR       126*   6207    6209    6257    6258    6264    6268    6808    6809
 COX         101*    654     747     748     909     920     936    1010    1082    1158    1227    1255    1308    1310    1402
            1419    1463    1599    1616    1634    3707    3709    3721    3723    3725    3727    3728    3902    3910    3914
            3939    3944    3945    4012    4013    4137    4390    5627    5637    5763    5773    5861    5883    6113    6143
            6483    6516    7580
 CPC         108*    604     606     803     805     839     841     852     854     869     871    1076    1078    1116    1119
            1160    1162    1166    1168    1181    1183    1233    1235    1259    1261    1264    1284    1357
 CPLUS      2247*   2640    6902    6702    6904    6904
 CQ         2318*   2541
 CR           35*     54*    395     757     879     888     905    1564    1623    2299    2302    3488    3637    3657    5614
            7290    8017
 CREL       4745    5260    5322*
 CRPRN      2280*   2947    2991    2994
 CRTGI        66*   8095    9860
 CSADR       105*   4263    4265    4372    4374    4386    4388    4408    4425    4428    4429    4431
 CSASM      1905*
 CSBUG      2204*
 CSC        2288*
 CSDIR       838     838     840     840    1906*   5821    5821    5823    5823
 CSEDIT      603     603     605     605     649     649     651     651    1810    2148*
 CSIMP      1942*
 CSMISC     2049*
 CSMNEM      778    1941*   5115    5161
 CSMULT     2061*   5412    6501    6501    6503    6503
 CSOP       2246*
 CSOPT      2129*
 CSQ        2272*   2626
 CSREL      2022*
 CURLN       128*   3373    3376    3386    3389    3408    3410    3442    3444    7403    7406    7460    7462    7463    7465
 CVAFP      5937    8117*
 CVFASC     8351*
 CVIFP      8520*
 CX         2238*   2990    3004
 CY         2314*   2996    3008
 [illegible]    11*     14     474    1298    2296    3297    3311    8197    9296
 [illegible]  1920
 [illegible]   406*   4057
 [illegible]    51*
 [illegible]   127*   4255    4257    4264    4293    4298    4327    4330
 [illegible]   136*
 DELC       3459    3464*   3469
 DELIM       342*   2311    5612    5624
 DELLN       955     961*   3468
 DFHMEM       31*   1025    1026
 DFLMEM       30*   1032    1033
 DGCNT       212*   8129    8159    8162    8171    8206    8209    8303    8307
 DIMP       5359    5367*
 DIRECT     2392
 DISASM     4556    5072    5096*
 DISFLG      177*   5043    5064    5074
 DIV        6937    6965*
 DMTBL      5361    5387*
 DMODE       348*   3332
 DMULTI     5362    5395*
 DOOPS      6182    6198*   6375
 DOREP      3806    3824    3832*
 DOSLOC       62*   3315
 DREL       5360    5377*
 DSADR       112*   4242    4244    4262    4273    4274    4279    4282    4284    4292    4297    4304    4325    4328
 DSPFLG       64*   3504    3553    7639    7852
 DSTBL      2356*
 DSYN2       560*    583     616    3334    4521    4537    4847    4849
 DSYNTA      565*    567    7875
 DXTBL      3040*
 ECOM        531     534*
 ECSIZE      117*   3116    3117    3121    3124    3152    3156    3162    3165    3221    3222    3247    3250
 EDERR      3462    7848*
 EDMSG       521     522     554*
 EEXP        203*   8125    8173    8222    8224    8231    8233    8236    8237    8240    8241    8250    8254    8258    8259
            8302    8316    8321    8438    8445    8452    8453    8464    8940    9462
 EMODE       347*   4440    6841
 END"       1925
 ENTDTD      137*   4103    4123    5687    5692    5831    5912    6787    6845    7528    7584    7652    7683    7702
 EOFFLG      250*   5641    5731    5836    6805
 EOP1       2439*   5107
 EOP2       2463*   5406
 EQULBL      364*    693    6331
 ERADR       460*   6630
 ERASYN      454*    736
 ERDUP       455*    713
 EREQU       457*   6356
 ERGARB      464*   5944
 ERLBIG      463*   1138
 ERLMEM      466*   4053
 ERLNUM      465*    992
 ERLOAD      452*   4109
 ERMSYN      451*   5650
 ERNDEF      453*   1373
 ERNLN       450*   3461
 ERNORG      467*   7116
 ERNSTR      459*   6127
```

```
EROVFL       468*     7470
EROVR        456*     1314      7106
ERPHAS       461*      705
ERRBUF       254*     7858      7861      7906
ERRDEX       136*     6403      6810      6848      7368      7854      7862      7888      7903      7905      7916
ERRMSG      7897      7898      7921*
ERRNUM       122*     7642      7656      7680      7706
ERROR        706       714       737       993      1139      1315      1374      3133      4054      4110      5651      5945      6128      6172      6357
            6632      7107      7117      7471      7672      7849*     7922
ERUFR        462*     6171
ERVAL        458*     7847
ESIGN        208*     8248      8273
ESTBL        603       605      2333*
ESYN2        515*      550       613      3818      4443      5750      7666
ESYNTA       524*      528       893       946       957      4067      7414      7877
ETXT         532       876*
EXBUFF        76*     4599      4612      4722      4725      4729      4732      4735      4757      4759      4762      4765      4768
EXINIT      4594      4719      4754*
EXP         9639*
EXP1        9652      9674*
EXP10       9644*
EXP2        9687      9689*
EXP3        9692*     9694
EXPAND      3115*
EXPERR      9643      9655      9663      9698      9699      9710*
EXPLOW      1056      1717      1729      3113*     5990      6781      6796      6800
EXPOUT      9703      9709*
EXPSGN      9682      9702*
EXPTXT       924      3107*     3873
EXTBL        649       651      3016*
FADD        8748*     9589      9734      9803      9822
FADR        4140      4966*
FAGET       4092      4884      4985*     5010
FAGET2      4153      4935      5009*
FASC        7956      8352*
FCHRFL       210*     8138      8190
FDIV        8974*     9708      9748      9782      9829
FHALF       9800      9801      9833*
FILEON       400*
FINDC       3774*     3797
FINDON       390*     3767
FLD01       9603*     9606
FLDOP       9602*
FLDOR       9573      9600*     9668      9707      9740
FLD11       9613*     9616
FLD1P       9612*
FLD1R       9588      9595      9610*     9641      9733      9743      9747      9798      9802      9828
FLIST       3515      5707      7571*
FLOAD       5685      7581*
FLOAD2      4060      4073      7579*
FLOG10      9783*
FLPTR       9600      9601      9603      9610      9611      9613      9620      9621      9624
FMOVE       9570      9632*     9665      9679      9704      9790      9804
FMOVE1      9633*     9636
FMOVER      3271*
FMPREC       189*     8154      8155      8761      8775      8792      8819      8843      8893      8908      8995      9007      9017      9050      9203
            9217      9272      9299      9349      9475      9494      9498      9502      9504      9507      9509      9513      9532      9548
```

```
 FNUL      8988*   9576    9642    9680    9701    9791    9799
 FOBJ      4138    5717    7589*
 FONE      9705    9706    9724*
 FPFP      9855*
 FPI        981    1440    8576*   9659
 FPORG       34*     58*   8104
 FPREC      188*    189     193     195     197     198    9201    9580    9602    9612    9622    9632    9654    9690    9722
           9777    9844    9856
 FPSCR     9656    9657    9666    9667    9735    9736    9745    9746    9787    9788    9796    9797
 FPT02     9564    9565    9571    9572    9579    9581    9583    9585    9586    9587    9726    9727    9731    9732    9741
           9742
 FRC        192*    194    1452    1454    3581    3584    6082    6184    6186    6230    6234    6236    6237    6363    6364
           6372    6374    6611    6612    6867    6868    6892    6893    6924    6926    6927    6929    6938    6940    6941
           6943    6950    6952    6953    6955    6969    6970    6984    7006    7007    7070    7079    7405    7408    7907
           7909    8324    8334    8336    8360    8409    8501    8525    8527    8546    8547    8548    8559    8586    8663
           8665    8684    8753    8764    8768    8785    8836    8838    8870    8941    8980    8995    9003    9004    9201
           9204    9223    9232    9237    9260    9302    9303    9308    9442    9449    9464    9494    9498    9550    9604
           9623    9633    9646    9649    9660    9662    9755    9763    9769    9806    9808    9813    9819    9820
 FRDEI      194*   6971    6972    6975    6978    6982    6983    7004    7005    7017    7018    7029    7030
 FROM       193*   8148    8154    8155    8795    8797    8813    8823    8825    8846    8847    9180    9207    9214    9215
           9230    9335    9344
 FRI        196*   6876    6877    6925    6928    6939    6942    6951    6954    6976    6979    7016    8742    8744    8750
           8765    8766    8786    8872    8879    8978    8986    9055    9264    9443    9447    9448    9465    9504    9534
           9614    9634    9685    9689    9692    9695    9700    9764    9776    9779
 FR1M       197*   8796    8824
 FR2        198*   9022    9176    9473    9509    9535
 FRA10     8902    9493*
 FRA1E     9070    9501*
 FRA20     8920    9497*
 FRA2E     9037    9506*
 FRE        195*   8893    8908    9021    9023    9054    9056    9502    9507    9551
 FRINIT    3770    3972*
 FRSIGN     204*   9445    9461
 FRX        199*   8214    8283    9190    9417    9471
 FSTO1     9623*   9626
 FSTOP     9622*
 FSTOR     9569    9620*   9658    9730    9737    9789
 FSUB      8741*   9669    9744
 GASML     5756    5867    5910*
 GCXI      7446    7486    7527*
 GETC1     3412    3724*   4046    4139    4254    4953    4957    5058    5971    6102    6423    6432    6474
 GETC2     3722*   3905    4009    4972    7606
 GETC3     3720*   5859
 GETCOD    3392    3398    3400    3403    3414    3509    3529    3532    3651    3668    3685    3688    3694    3698    3701
           3703    3726*   3977    3980    4048    4061    4234    4241    4243    4248    4256    4381    4385    4387    4394
           4406    4409    4775    4778    4780    4955    4967    4969    4974    5045    5049    5051    5054    5060    5676
           5698    5703    5708    5712    5774    5817    5875    5962    5999    6086    6096    6106    6119    6364    6386
           6399    6476    6497    6506    6508    6519    6871    7043    7045    7056
 GETHEX    1464*   5619
 GETIN     1406    1528    3739*   5611    5623    7286    7309
 GETLL      918     962    3365    3450    3547    3795    3933    5932    7819    7824*
 GETSTM     913     953    3351    3433    3711    7410    7793*
 GJADR     4668    4676    4790*
 GJADR1    4678    4792*
 GLGO      3483    3815    5917    7485*
 GLPCX     5025    5688    7531*   7685
 GLPCX1    7532*
```

```
GLPX      7500      7533*
GNLINE    1004      7481*
GNXTL      932       964      3366      3451      3548      3796      5933      7820      7829*
GNXTL1    3862      7831*
GOBJ      4075      4078      4083      4085      4087      4089      4094      4122*
GPCHAR    4125      4158      5024*
HEXTAB    7992      8000*
HIMEM       92*     1027      1028      3125      3128      4033      4034
HMADR       29*       57*       92
IASC      3588      7930      7954*
ICAUX1     292*     7502      7503      7611
ICAUX2     293*     7504      7505      7760
ICAUX3     294*     5694      7690
ICAUX4     295*     5696      7692
ICAUX5     296*     7688
ICBAH      288*     7754
ICBAL      287*     7756
ICBLH      291*     7749
ICBLL      290*     7751
ICCLOS     315*     7600      7715
ICCOM      285*     7758
ICDDC      317*
ICDNO      284*
ICFREE     319*
ICGBC      309*     4124
ICGBR      307*
ICGTC      310*
ICGTR      308*     7529
ICHID      283*
ICLEN      298*      300
ICMAX      318*
ICOIN      304*
ICOIO      306*     7603
ICOOUT     305*
ICPBC      313*     4157
ICPBR      311*
ICPTC      314*
ICPTR      312*
ICPUT      289*     7518      7520
ICSBRK     326*     7644
ICSDER     329*
ICSDNR     327*
ICSEOF     325*
ICSIVC     330*
ICSIVN     332*
ICSNED     328*
ICSNOP     331*
ICSOK      323*
ICSTA      286*     7723
ICSTAT     316*
ICSTR      324*
ICSWPE     333*
IDISPL     268*      760      5761      7263
IF        1907
IFP       7955      8521*     9664      9815
ILEN       244*     4595      5125      5148      5276      5306
ILSHFT    1499      8712*
```

```
IMM      6637*  6653   6689   6699   6702   6711   6714   6727
IMMEDI   2292
INBOUT   3563   3627*  5797
INBUFF   2179*   668    756    878    887    902   1251   1324   1543   1622   1707   3487   3592   3595   3615
         3631   3741   3822   4128   4999   5878   6289   6292   6298   7420   7422   7425   7434   7436   7437
         7439   7752   7753   7937   8494   8496   8497   8499   8507   8704   8706   9093   9108   9120   9426
         9428   9429   9431
INCLA    5242   5250   5303*
INCMVF   4097   4913   5011*
INOPST   3749   3752   3763   3903   4008*
INCSRC   1877   1888*  5180
INDX     6641*  6653   6725
INDY     6642*  6653   6726
INTLBF   1002   3575   3587   5911   7450   8353   8702*
INVOPT   5420   5476*  5494
IO1      7487   7612   7745*
IO2      7747*
IO3      5028   7448   7746   7748*
IO4      7749*
IO5      7752*
IO6      7754*
IO7      5689   7693   7757*
IO8      7601   7716   7758*
IOCB      282*   298   7503   7505
IOCBOR     26*    50*   275
IOCMD     251*  7531   7604   7757
IODVC     252*  4105   7534   7597   7619   7651   7735   7738
IOTES2   7514   7632*
IOTEST   5029   5690   7449   7488   7613   7630*
ITTAB    5346   5348   5358*
KWSRCH    539    572    762   1338   1772*
KWSTRT    340*  1786   1901   2474   2475
KWTAB    1775   1776   1900*
KWTAB2   1944*  5164   5165
L        6106*
LBLS1    1688*  6295
LBLSRC    674   1365   1657*
LBPR1     234*
LBPR2     235*
LBUFF     236*  1736   1860   3652   3658   4882   4887   4899   4908   8356   8377   8379   8393   8414   8420
         8421   8422   8423   8487   8703   8705   9374   9388   9402
LCHK1     343*  4076   4143   5720
LCHK2     344*  4079   4142   5719
LDDVX    5693   7535   7598   7609   7618*  7713   7722
LDIOST   7631   7694   7721*
LEADR     107*  5059   5061   5083   5087
LELNUM    130*  3677   3678   3687   3690   3702   3704   4210   4215
LENTAB   6617   6756*
LGCOEF   9793   9794   9834*  9844
LINDEX    179*  3782   3828   3861   3885   3888   3889   3935   3940   3952   3953   3961
LINLNG    123*  3843   3934   3954
LISDV     376*  7573
LIST     3481   3501*
LISTI    3544   3561*  3790   3811
LISTC    3537*  3549
LISTDT    138*  3521   3525   6786   6846   7495   7574   7703
LMADR      28*    56*  1034   1035   1037   1039
```

```
LMBAV        256*   1038    1040    1043    1044    4047    4049
LNCNT        249*   5040    5077    6029    6807    7382    7383
LNINCR       139*   3372    3375    3387    3390    3399    3401    3407    3409    3413    3415    7461    7464
LNUMDF       357*   3479
LNUMDN       356*   3498    3993
LOADDV       378*   7583
LOCCNT       142*    682     685     695     699    1422    1424    5632    5635    6185    6187    6262    6266    6371    6373
            6437    6440    6803    6804    7129    7131    7132    7134    7145    7147    7149    7160    7167    7169    7177
            7179    7227    7228    7239    7241
LOG         9751*
LOG1        9763*
LOG10       9753*
LOG10E      9639    9640    9723*   9826    9827
LOG2        9771    9774*
LOG3        9773    9776*
LOG4        9779*   9781
LOG5        9756    9759*
LOG6        9809    9814*
LOG7        9817    9821*
LOGBTH      9752    9754*
LOGDON      9825    9830*
LOGERR      9757*
LOGOUT      9823*
LOMEM         74*   1045    1046    4021    4022
LOMFLG       132*    500     549    4043
LONE        5532*   5546
LRANG1      3458    3667*   3973
LRANGE      3534    3669*
LSADR        114*   4552    4554    4631    4633    4843    4846    5050    5052    5082    5086    5100    5135    5256    5271
            5274    5292    5293    5305    5307    5308    5310    5325    5328    5336    5339
LSEARC      3784    3932*
LSYNER      5766    5787*
LTHREE      5534*   5543    5544    5545
LTWO        5533*   5537    5538    5539    5540    5541    5542
LTYPFL       174*   3480    3499    3576    3994    5927
MAMODE       350*   1659    5602    5789    6623    7200
MASYNT      5604*   5608    5640    5642    5654
MAXCIX       111*   1009    1219    1221
MDEND       8939*   9076
MDESUP      8875    8983    9441*
MDSUP       8884    8991    9460*
MEMFUL       449*   3132
MEMTOP        89*   1068    3120    3123    3139    3142    3169    3174    3225    3232    3254    3258
MEND1       8944*
MEND2       8873    8949*
MEND3       8871    8951*   8981
MISCRA        23*     45*    229
MODBYT      6580    6739*
MODEFL       135*   1658    3333    4441    5603    5670    5788    6622    6842    7199    7660    7664    7866    7871    7890
MOVLIN      3562    3572*   5931
MOVNUM      6866    6875    7041*
MOVTXT      3577    3602*
MSGADR       103*   8067    8068    8075
MSGXY       7899    8066*   8071
MSGXYC       523     564     628    6037    8090*
MUL         6916    7000*
MVOTO1      9631*
```

```
 MVFA       151*  3147  3150  3185  3188  3193  3238  3241  3280  3283  4084  4086  4096  4146  4147
           4895  4896  4968  4970  4986  4990  4998  5012  5014
 MVFEA      116*  4088  4090  4149  4150  4973  4975  4987  4991
 MVFROE    9478  9547*
 MVFR12    9468  9531*
 MVLNG      115*  3141  3144  3157  3178  3180  3231  3234  3237  3265  3272  3274
 MVTA       152*  3153  3158  3186  3189  3194  3243  3262  3266  3269  3281  3284  4881  4904  4905
           4916  4918  4941  4943  4945  4954  4956

LINES PROCESSED=ALL
UNUSED SPACE =  1978
```

```
A
Y
1
                                END OF EXCLUDES
                                END OF INCLUDES
NOPQRSTUVWXYZ
. +-*/,()=#@&
Y YYYYYYYYYYYYYYYYYYYYYYYYYYYYYYY
72                              LAST COLUMN CARD
0300


NATCF      9856*
NIBSHO     8151     8319     9179*    9415
NIBSH2     9175*    9470
NLCOEF     9792     9844*
NOCDD      7714     7724*
NOINV      5418     5494*
NORM       8326     8561     8804     8814     8831     8852     9199*
NORM1      8945     9202*
NOROP      4572     4591*
NOROP1     4593*    4815     4817     4819
NPCOEF     9675     9722*
NSIGN       205*    8202     8331
NTCMD       335*    5686
NULFLG      144*    1560     1579     1590     1597     1637
NZ          363*    1384
OBJBUF       77*    1050     7100     7155     7237
OBJDEX      109*    5771     6618     7099     7101     7130     7143     7173     7209     7230     7244
OBJDTD      182*    4156     4163     6220     6788     7156     7556     7592     7704
OBJDV       377*    7591
OBJEND       78*    1052     1054
OBJM2      6084     6438     6613     6627     7091*
OBJMAN     6107     6116     6424     6433     6445     6486     6609     6628     7092     7098*
OCNTL      5782     7124*
OERR        370*    6415     7216     7366
OFF        6366     6385*
OLBCNT      121*    5781     7232
OLIST       368*    6417     7363
ONEBYT      411*    6123     6425
OOBJ        369*    6178     6414     7138
OPAGE       371*    6416     7388
OPATYP     6501     6503     6655*
OPEN       7575     7585     7595*
OPERAN     2292
OPSRCH     5108     5407     5505*
OPSYN       810*     842      856      873
OPT"       1928
OPTFLG      271*    6177     6381     6362     6395     6396     6806     7137     7215     7362     7365     7387
OPTTAB     6379     6394     6413*
ORRES      7042     7054*
OSRCH      6204     6216     6282*
OUT        5780*    5858
OUTBUF       75*     741      910      938     1309     1641     3729     3911     3949     5631     5634     6310     6314     6317
OUTCNT      161*    3574     3590     3614     3616
P1OCOF     9676     9677     9712*    9722
```

```
PABS       5389    5543*   5572    5573    5574
PABSX      5449    5544*   5588    5589    5590
PABSY      5451    5545*   5585
PACU       5546*   5570
PAGE*      1913
PAGEX      6018*   7393
PAGEND       88*   6778
PAGEP        87*   6015    6049    6050    6814    6818
PASS1       361*   5671
PASS2       362*   5599    5726
PASSFL      133*    717    1078    5600    5672    5734    5737    6199    7125    7359    7678
PEJECT      341*   6021
PIMM       5537*   5560    5562    5569
PIN        7203    7254*
PINDX      5538*   5561
PINDY      5539*   5577
PIOV4      7857*
PLST        414*   5795    6140    6822    6824    6826    7258
PLYARG     9567    9568    9593    9594    9728    9729    9738    9739
PLYCNT      207*   9566    9574    9591
PLYEV1     9576*   9596
PLYEVP     9582    9586*
PLYEVL     9564*   9678    9795
PLYOUT     9575    9577    9590    9592    9597*
PMASK       247*   5126    5205    5212    5223    5225
PMTAB      5427    5559*
PNUM       3343    3384*   3426
POBJL      7195    7211    7225*
PORG       5834    6180    6256*   6390
POX         272*   6341    7188    7325    7391    7395    7498
PRA        5206    5237*
PRADR      5221    5270*
PRADRB     4275    4897    4906    5294    7229    8028*
PRBLAN     5155    5796    7328    7342*
PRBLK1     7345    7911    8007*   8030    8043
PRBLK3     4025    4031    4337    4902    5201    5295    7340*
PRCHAR      625    3633    3814    5197    5231    5235    5239    5253    6022    7288    7310    7429    7493*   7895    7940
           7993    8009    8018    8053    8055    8079
PRCNT       160*   5215    5217    5219    5232    7304    7311
PRCR        520     561    3507    4037    4271    4534    4910    5243    5265    6023    6024    6025    6026    6033    7207
           7917    8016*   8092
PRCX       7497*
PRHEX2     4023    4029    4035    5264    6345    7964*   8029
PRHEXB     4280    4341    4346    4351    4356    4361    4900    4909    5136    8041*
PRHEXC     4339    4344    4349    4354    4359    8052*
PRIHEX     5281    5284    7238    7767    7978*   8042
PRINUM     7417    7910    7929*
PRPAD      7259    7267    7274    7281    7323*
PRPAGE     6032    6048*
PRPC       4851    5097    5291*
PRPL       7264    7272    7279    7301*
PRREL      5207    5251*
PRTERR     3134    5653    5946    6173    7118    7218    7870    7887*
PRTITL     6031    6043*
PTBUFF      102*   5967    5969    5975    5998    6006
PTCMD       334*   7684
PTDEX       169*   5996    6000    6001
```

```
PTSTR     5961*   6016
PUTC1     7164    7496*   7557
PUTCHA    7494*
PUTO2     4144    4148    4151    5721    6238    6245    7549*
PUTOBJ    7552    7555*
PZFQ      5540*   5564    5565    5566
PZFQX     5541*   5580    5581    5582
PZFQY     5542*
QTEMP     8995*   9030    9041    9042    9043    9044    9063
QUERRY    3810*
RADFLG     157*
RAM2        24*     46*    242
REPFLG     181*   3769    3787
REPON      389*   3755
RESBYT     173*   5851    6167    6328    6521    6864    7061    7062
ROM         32*     52*    473
RSHF0E    8930    9075    9298*
RSHFT0    8810    9259*
RSHFT1    8779    9263*
RSTEP     4614*   4764    4766
RSTEP0    4624*   4852
RSTEP1    4636*   4670    4679    4707    4749
RSTEP2    4559    4630*
RSTVB     4491    4839    4858*
RTNADR     127*   4546    4548    4640    4642    4848    4850
S         7472
S1         475*
S2         476*
SCRDV      375*   3524    7701
SCVECT    8096*   8101    9861*   9866
SDEND     4232    4246    4323*
SEARCH    1692    1777    1844*
SET1       385*   3989
SETA       383*   3985
SETBYB      48*   4460    4862
SETQ       384*   3981
SETUPC    4449    4519    4531    4774*
SGNFLG     209*   9647    9702    9754    9824
SHEP         1
SICKIO    7633    7636*
SIX        230*   1085    1157    1224
SKBLAN    1013    1435    1561    9103*
SKCTL       65*    518
SKPBLA     535     595     754     782     820    1418    1466    1661    1773    5610    5622    8119    9104*
SMBLD       84*   5739    5740    6779
SMBLEN      83*   1716    1728
SMBLTP      82*   1689    1690    6784    6851
SOX        231*   1083    1159    1226
SPC        232*   1077    1079    1161    1163    1180    1182
SPTAB     4582    4584    4804*
SQRIO     9784    9785    9802*
SQVCNT     205*
SRCADR     104*    679     684     687     697     701     719     725     727    1369    1383    1385    1391    1394    1850
          1851    1855    1862    1871    1873    5168    5169    5176    5194    5507    5508    5513    6206    6208    6229
          6233    6241    6244    6302    6315    6318    6322    6325    6330    6332
SRCCNT     124*   5506    5517
SRCNST    1698    1876*
```

```
SRCBKP    125*    1845    1854
STACK     229*     233
STENUM    176*     699     769     835     864    1343    1784    1787    1848    1852    4557    5116    5159    5412    5469
         5460
STEP     4520     4533    4544*
STKLVL    119*    1081    1142    1154    1175    1189    1223    1257
STMCVR     80*     926     928     939     967    3111    3217    3360    3363    3443    3446    3580    3583    3604    3613
         3847     3851    3853    3894    3896    3913    3957    4189    4194    4209    4214    5665    5667    6832    6834
         7797     7798    7805    7810    7826    7832    7833    7835    7837
STMSTR    180*     537     570
STOP      167*    5101    5396    5402    5444    5511    6507    6585    6608
STR1     5230     5553*
STR1LN    118*    3751    3764    3836    3996
STR2     5233     5554*
STR2LN    267*    3753    3838    3860
STRDEL    175*    1606    1625
STRDEX    246*    3946    4000
STRMOV   3647*    7607
SUB      6915     6948*
SVCIX     159*    5870    5894
SVESA     106*     925     927    1723    1737    1744    1746    3151    3155
SVLENG    164*    5860    5871
SYNINT    527      566    1001*   5607
TAB'     1931
TAB1      178*     243*   6142    6823    7266
TAB2      245*    6825    7273
TAB3      248*    6827    7280
TABCNT    168*    6136    6141    6146
TABEXP   6902     6904    6913*
TABS      442*    3010    6531
TABSX     444*    3002
TABSY     443*    3006    6571    6600
TACU      438*    2981    6512    6540
TADR     4869     4928    4952*
TCOMM     425*     740
TDIR      426*     771
TEMPRL    258*    5695    5697    7689    7691
TEOE      422*    2603    6872
THEX2     419*    1377    1386    1403    1420    1447    1509    3682    3699    4382    6098    6510
THRBYT    412*    6490
TIMM      437*    2970    6542    6591
TIMP      427*
TINDX     435*    2988
TINDY     436*    2993
TITLE'   1910
TITLEN     86*    6794
TITLEP     85*    5738    5955    6044    6045    6813    6817
TLBL      423*     723
TLEND    3465     3541    3778    4176*
TMACRO    431*
TMISC     429*
TMULTI    430*    5110
TNLBL     424*     745
TOBJDE    163*    7153    7171    7172    7187    7236    7243
TOPT2    2440*    2463    5404    5405
TOPTBL    778     2401*   2439    5105    5106
TPCIX     170*    7419    7427    7435    7443    7933    7935    7936
```

```
TPV01    4483   4488*
TPVBLK   4457   4458   4481*
TREL      428*
TSCDX     172*  1617   1635   1640   5862   5882
TSLNUM    113*   987    988   3349   3350   3361   3364   3371   3374   3431   3432   3674   3675   3686   3689
         7404   7407   7806   7811
TSOE      421*
TSTALR   1322*  5889
TSTBRK   1015   3484   3538   3545   3775   3816   4289   4392   4535   4877   4933   5069   5726   7769*
TSTEJT   5794   6338   7193   7381*  7892
TSTLIS   5792   6019   6336   7190   7358*
TSTLST   4179   4207*
TSTNUM   1674   5891   9089   9118*
TSTOER   5799   7214*
TSTR      420*  1610
TSTTEN   3356   3438   4177   4188*  5922   7802
TVSCIX    171*  1437   1443   1663   1684   1686   1696   1702   1734   1749   3908   3912   3915   3941   3959
         6294
TWOBYT    410*  6089   6465
TXTDV     379*
TXTEND     81*  4027   4028   4190   4195   6852   6853
TXTTAB     79*  5664   5666   6831   6833   7794   7795
TYPEN    6653*  6657   6660   6663   6666   6669   6672   6675   6678
TYPEN2   6654*  6681   6684   6687   6690
TZPG      439*
TZPGX     441*
TZPGY     440*  6569
UPAUTO    944   7458*
UPC       140*  4474   4551   4553   4562   4598   4632   4634   4654   4657   4701   4706   4713   4742   4746
         4747   4779   4781   4793   4796   4797   4798
UPCADR   4401   4421*
UPCUR    3448   7457*
URA       260*  4340   4470   4610   4618   4834
UREG      259*   260    261    262    263    264   4371   4373
URP       263*  4355   4468   4608   4623   4688   4838
URS       264*  4360   4465   4605   4626   4650   4662   4685   4690   4696   4704
URX       261*  4345   4467   4607   4619   4830
URY       262*  4350   4464   4604   4620   4832
VBRK     3324   3326   4828*
VERROR   4416   6463   7077   7846*
WAITCR   3482*  3489
WARMFL     61*   485
WARMST    476    486    509*
WHAT      612    615    623*
WHTMSG    626    627    632*
WORD"    1916
WORDFL    120*  6073   6078
WORDON    405*  6071
XA       3041   5595*
XASM     3032   5660*
XATBUG   3033   3322*
XBR      4716*  4806   4808   4810   4812   4814   4816   4818   4820
XBR1     4730   4733   4740*
XBRK     4712*  4805
XBYE     3030   3296*
XBYTE    3075   6095*  6121
XC       3052   4380*
```

```
XCLR      1042*   3018    4054
XCR       3043    4270*
XD        3048    4226*
XDBYTE    3076    6066*
XDEL      3019    3457*
XDOS      3031    3310*
XDW       3044    4336*   4636
XEND      3077    5830*
XENDC     5757    5833*   5905
XENTER    3021    4059*
XEQU      3081    6308*
XEXP      5846    6075    6114    6138    6164    6320    6435    6487    6518    6862*
XFIND     3020    3762*
XFMFLD     211*   9645    9661    9681    9768    9774    9807    9816
XFORM     9726*   9786
XH        3045    4448*
XIF       3071    5845*
XIJMP     4566    4675*
XITVBV      49*   4484    4859    4860
XJMP      4564    4667*
XJSR      4649*   4807
XL        3049    5034*
XLIST     3017    3497*
XLOAD     3022    4072*
XLOMEM    3029    4042*
XM        3042    4927*
XNEW      1041*   4050
XNUM      3024    3339*
XOPT      3078    6363*   6401
XORG      3080    6163*
XPAGE     3073    6014*
XPRINT    3025    3478*
XREN      3026    3425*
XREP      3027    3748*
XRTI      4684*   4809
XRTS      4695*   4811
XS        3047    4518*
XSAVE     3023    4136*
XSIZE     3028    4020*
XT        3046    4530*
XTAB      3079    6134*
XTITLE    3072    5953*
XU        3050    4868*
XWORD     3074    6070*
XX        3051    4439*
ZF1       8686*
ZFP         33*     53*    166     187
ZFRO      8131    8530    8683*   8950    9245    9479
ZICB        19*     41*   7503    7505
ZPG       6639*   6653    6654    6693    6696    6699    6702    6705    6708    6711    6714    6717    6729
ZPG1        20*     42*     73
ZPG2        21*     43*    150
ZPG3        22*     44*    153
ZPGX      6643*   6653    6654    6693    6696    6702    6708    6731
ZPGY      6646*   6698    6704    6730
ZT1        145*   3648    3654    3842    3845    3866    3884    3887    5130    5138    5149    5152    5162    5171    5190
          5193    5198    5335    5337    6502    6561    7233    7246    8071    8073    8074
```

```
ZT2           273*  5132  5134  5137  6504  6564
ZTEMP1        221*  8600  8642  8895  8900  8913  8918  8934  8992  9072  9476
ZTEMP3        223*  8612  8614  8623  8626  9267  9282
ZTEMP4        222*  1471  1472  1483  1484  1485  1487  1511  1513  8526  8528  8537  8538  8581  8582
             8611   8613  8622  8624  8625  8627  8635  8636  8637  8640  8652  8654  8655  8657  8662
             8664   8715  8716  8752  8756  9268  9269  9283  9290  9319  9331  9355  9514  9523
ZXLY         8127   8688*

LINES PROCESSED=ALL
UNUSED SPACE =  9932
```